Пермский филиал федерального государственного автономного образовательного учреждения высшего образования

«Национальный исследовательский университет

«Высшая школа экономики»

Факультет экономики, менеджмента и бизнес-информатики

Образовательная программа бакалавриата «Программная инженерия»

**ОТЧЕТ**

**по проектной работе**

Выполнил студент группы ПИ-20-2

Копылов Никита Евгеньевич

(подпись)

Руководитель проекта

Старший преподаватель

информационных технологий в бизнесе

Вячеслав Владимирович Ланин

\_\_\_\_\_\_\_\_\_\_\_

(оценка) (подпись)

(дата)

Пермь, 2021

**Оглавление**

[Введение 3](#_Toc75895297)

[Аннотация 4](#_Toc75895298)

[1 Назначение разработки 5](#_Toc75895299)

[2 Постановка задачи 6](#_Toc75895300)

[Общая 6](#_Toc75895301)

[Данного студента 6](#_Toc75895302)

[3 Анализ 8](#_Toc75895303)

[4 Анализ аналогов 9](#_Toc75895304)

[5 Код программы 11](#_Toc75895305)

[6 Руководство пользователя 19](#_Toc75895306)

[6.1 Область применения 19](#_Toc75895307)

[6.2 Краткое описание возможностей 19](#_Toc75895308)

[6.3 Уровень подготовки пользователя 19](#_Toc75895309)

[6.4 Рекомендации по освоению 19](#_Toc75895310)

[7 Проектирование 20](#_Toc75895311)

[8 Тестирование 21](#_Toc75895312)

[9 Заключение 25](#_Toc75895313)

[10 Список используемых источников 26](#_Toc75895314)

Введение

Тема курсового проекта: «Разработка чат-бота для игры в города на языке C#» отчет выполнен на 26 страницах, содержит 7 рисунков, 10 использованных источников. Необходимость выполнения настоящего проекта обусловлена потребностью пользователей получения развлекательного и образовательного контента, путем игры в Города, в мессенджере Telegram, а также отсутствием приложений, которые бы предоставляли возможность игры в города без скачивания программы.

Цель работы: целью проекта является изучение особенностей конструирования и применения чат-ботов, разработка чат-бота для игры в города в мессенджере Telegram. Для выполнения поставленной цели необходимо решение следующих задач:

* изучить понятия чат-бота и его функций;
* рассмотреть преимущества использования мессенджеров для игры в Города;
* рассмотреть языки программирования и выбрать язык для разработки;
* выбрать мессенджер, в котором будет реализован чат-бот;
* разработать ИТ-проект по разработке чат-бота;
* продемонстрировать концепцию разработки.

Объектом исследования данной работы является программирование на языке C# для создания чат-бота, который позволит пользователям играть в Города прямо в мессенджере Telegram. Предметом данной работы является чат-бот, который позволит играть в Города прямо в мессенджере Telegram. Практическая значимость заключается в том, что данный чат-бот позволяет играть в города без скачивания каких-либо приложений и получать дополнительную информацию по данным городам.

Аннотация

Курсовой проект посвящен разработке чат-бота, который позволит пользователю играть в Города, в мессенджере Telegram. Подробнее рассмотрена часть работы, которая выполнена автором отчета. Отчет состоит из семи частей. В первой части будет рассмотрено назначение разработки, во второй части будет произведена постановка задачи общая и для конкретного студента, в третьей части будет произведена аналитика целей проекта, в четвертой части будет проведен анализ аналогов и выявление преимуществ нашего продукта, в пятой части будет приведен код части программы, над которой работал данный студент, в шестой части приведено руководство пользователя, в седьмой части отчет производится проектирование работы и взаимодействий программы, в восьмой части произведено тестирование части программы, которую разрабатывал студент на предмет ошибок или некорректных ответов на запросы пользователя. В заключительной части будет сделан вывод о том, какая работа была проделана, какие особенности получил продукт разработки, что вызвало наибольшее затруднение, а что, наоборот, было простым и быстро удалось реализовать.

1. Назначение разработки

Данный студент занимался частью разработки, которая посвящена реализации дополнительных возможностей чат-бота (геолокация города, ссылки на Яндекс и Гугл с запросом о городе, справка из Википедии, показ города на карте). Реализация дополнительных возможностей происходила посредством подключения встроенных кнопок, которые имеют надписи, соответствующие назначению кнопки (например: кнопка “информация о городе” предоставляет информацию из Википедии о городе).

1. Постановка задачи

Общая

В ходе выполнения проекта необходимо:

* проанализировать и выбрать язык программирования, на котором будет написана программа, реализующая работу чат-бота;
* проанализировать мессенджеры и выбрать тот, который лучше подходит для реализации чат-бота;
* написать программу для бота;
* реализовать возможность предоставления пользователю геолокации города;
* реализовать возможность предоставления пользователю ссылки на справки из Википедии по городу;
* реализовать возможность предоставления пользователю координат города на карте;
* реализовать возможность предоставления пользователю ссылки на фотографии города;
* реализовать возможность предоставления пользователю ссылки на Яндекс или Гугл с запросом по городу.

Данного студента

В ходе выполнения проекта со стороны студента необходимо:

* реализовать возможность предоставления пользователю геолокации города;
* реализовать возможность предоставления пользователю ссылки на справки из Википедии по городу;
* реализовать возможность предоставления пользователю координат города на карте;
* реализовать возможность предоставления пользователю ссылки на фотографии города;
* реализовать возможность предоставления пользователю ссылки на Яндекс или Гугл с запросом по городу.

1. Анализ

Для реализации проекта был выбран язык программирования C#, так как на этом языке разработчики имеют наибольший опыт работы, также была выбрана среда разработки Microsoft Visual Studio, так как она имеет множество поддерживаемых библиотек, возможность одновременной работы нескольких пользователей (Live Share) и функцию подсказки синтаксиса. Для реализации Inline кнопок были использованы встроенные классы InlineKeyboardButton и TelegramBotClient и их методы WithCallbackData, SendTextMessageAsync, SendChatActionAsync, SendPhotoAsync, WithUrl. С помощью метода WithCallbackData выводится текст в виртуальную кнопку и связи с ссылкой на ту или иную страницу, в зависимости от назначения кнопки. Метод SendTextMessageAsync использовался для отправки сообщений пользователю с ответом на его сообщение. Метод SendChatActionAsync используется для уведомления пользователя о том, что при длительной неактивности бот не перестает работать, а проводит какой-либо процесс (например, если бот долго грузит фотку, то появляется надпись о том, что фотография загружается). Метод SendPhotoAsync служит для отправки фотографии в чат, в нашем случает с помощью этого метода отправляются фотографии городов из Википедии. Метод WithUrl служит для создания информационнной составляющей виртуальной кнопки (то бишь, что будет написано внутри нее) и для привязки ссылки к этой кнопке.

1. Анализ аналогов

Анализ рынка программных продуктов позволил выделить три основных программы, используемых для игры в Города. К ним относятся такие программные продукты как «Игра в города» - бот в мессенджере Telegram, «Сыграем в Города» - приложение из Google Play для Android, «Города- Игра от А до Я» приложение из App Store. Рассмотрим особенности каждой из программ и выделим ее сильные и слабые стороны.

«Сыграем в Города» является приложением, предназначенным для Android. Для игры данное приложение нужно скачать бесплатно из Google Play. К плюсам данного приложения можно отнести удобный интерфейс, возможность поменять тему, а также возможность получать достижения в Google Play. К минусам - встроенную рекламу, долгие ответы приложения (примерно 10-15 секунд) и невозможность узнать дополнительную информацию о городе (предоставляется только название города и флаг страны, в которой он находится).

«Города- Игра от А до Я» является приложением, предназначенным для iOS. Для игры данное приложение нужно скачать за $0.99 из App Store. В программе предусмотрены три режима: классический, на время и пятиминутка, в каждом из которых реализован ввод города с уже написанной первой буквой, что безусловно является плюсом. Также в приложении предусмотрен удобный набор очков, позволяющих пользователю взять подсказку. К недостаткам данного аналога можно отнести встроенную рекламу, отсутствие возможности играть с другими пользователями и отсутствие подсказок для игрока.

«Игра в города» является ботом в мессенджере Telegram. Для игры пользователю нужно иметь доступ в мессенджер Telegram. В данном боте эргономично расположены правила игры в Города, условия пользования ботом и его возможности, сделано это с помощью команд, которые пользователь может вызвать даже в момент игры. Недостатки данного бота заключаются в невозможности начать новую игру, кроме того, игра происходит одновременно со всеми участниками, подписавшимися на бота.

Приведенный анализ показал, что все из рассмотренных программ имеют низкие требования к системным ресурсам, однако все из них требуют скачивания. Также во всех приведенных программах нет возможности получить дополнительную информацию о городе, что с положительной стороны выделяет наш продукт из всех аналогов.

1. Код программы

private static void Bot\_InlineKeyboardButton(object sender, CallbackQueryEventArgs e)

{

try

{

var s = e.CallbackQuery.Data.Split("|");

if (s.Length > 0)

switch (s[0])

{

case "More":

Bot\_MoreCityInfo(s, sender, e);

break;

case "NextLetter":

Bot\_NextLetter(s, sender, e);

break;

case "Restart":

Processing.DeleteUser(s[1]);

Bot\_StartCommand(sender, new MessageEventArgs(e.CallbackQuery.Message));

break;

}

}

catch (Exception exception)

{

var s = $"Bot\_InlineKeyboardButton - e.Message.Text: {e.CallbackQuery.Message.Text} - e.Message.Chat.Id: {e.CallbackQuery.Message.Chat.Id} - {exception}";

Logging.ERROR(s);

}

}

public static void Main()

{

\_botClient = new TelegramBotClient(Configuration.BotToken);

\_me = \_botClient.GetMeAsync().Result;

Console.WriteLine($"Hello, World! I am user {\_me.Id} and my name is {\_me.FirstName}.");

\_botClient.OnMessage += Bot\_OnMessage;

\_botClient.OnMessageEdited += Bot\_OnMessage;

\_botClient.OnCallbackQuery += Bot\_InlineKeyboardButton;

Processing.TelegramUserTimerMinute += EveryMinuteEvent;

\_botClient.StartReceiving();

while (Console.ReadLine() != "exit") { }

\_botClient.CloseAsync();

}

private static async void Bot\_Get(object sender, MessageEventArgs e)

{

if (e.Message.Text != null)

{

string city = e.Message.Text;

Processing.Get(ref city, "Telegram." + e.Message.Chat.Id.ToString(), out bool onLastLetter, out bool cityIsUsed, out string outCity,

out char nextLetter, out byte letterNumberFromEnd, out string wikiUrl, out string yandexUrl, out string googleUrl, out string mapUrl,

out (decimal latitude, decimal longitude) coordinateCity, out string photoUrl);

Logging.DEBUG($"Telegram: Chat: {e.Message.Chat.Id}, Text: {e.Message.Text} " +

$"onLastLetter {onLastLetter}, cityIsUsed {cityIsUsed}, outCity {outCity}, nextLetter {nextLetter}, letterNumberFromEnd {letterNumberFromEnd}, " +

$"wikiUrl {wikiUrl}, yandexUrl {yandexUrl}, googleUrl {googleUrl}, mapUrl {mapUrl}, coordinateCity.latitude {coordinateCity.latitude}, " +

$"coordinateCity.longitude {coordinateCity.longitude}, photoUrl {photoUrl}");

if (cityIsUsed == true)

{

try

{

if (letterNumberFromEnd != Processing.DatabaseUsers["Telegram." + e.Message.Chat.Id.ToString()].OutCity.Length - 1)

{

await \_botClient.SendTextMessageAsync(

chatId: e.Message.Chat,

text: $"Город {e.Message.Text} уже был использован, попробуйте ввести другой город на эту же букву",

replyMarkup: new InlineKeyboardMarkup(

new List<List<InlineKeyboardButton>>

{

new List<InlineKeyboardButton>

{

InlineKeyboardButton.WithCallbackData("Попробовать на следующую букву",

$"NextLetter")

},

})

);

}

else

{

await \_botClient.SendTextMessageAsync(

chatId: e.Message.Chat,

text: $"Город {e.Message.Text} уже был использован, попробуйте ввести другой город на эту же букву"

);

}

}

catch (Exception exception)

{

var s = $"CityIsUsed - e.Message.Text: {e.Message.Text} - e.Message.Chat.Id: {e.Message.Chat.Id} - {exception}";

Logging.ERROR(s);

}

}

else if (onLastLetter == false)

{

try

{

if (letterNumberFromEnd != Processing.DatabaseUsers["Telegram." + e.Message.Chat.Id.ToString()].OutCity.Length - 1)

{

await \_botClient.SendTextMessageAsync(

chatId: e.Message.Chat,

text: $"Город {e.Message.Text} начинается не на последнюю букву предыдущего города. Ваша буква {nextLetter}.",

replyMarkup: new InlineKeyboardMarkup(

new List<List<InlineKeyboardButton>>

{

new List<InlineKeyboardButton>

{

InlineKeyboardButton.WithCallbackData("Попробовать на следующую букву",

"NextLetter")

},

})

);

}

else

{

await \_botClient.SendTextMessageAsync(

chatId: e.Message.Chat,

text: $"Город {e.Message.Text} начинается не на последнюю букву предыдущего города. Ваша буква {nextLetter}."

);

}

}

catch (Exception exception)

{

var s = $"NotOnLastLetter - e.Message.Text: {e.Message.Text} - e.Message.Chat.Id: {e.Message.Chat.Id} - {exception}";

Logging.ERROR(s);

}

}

else if (outCity == null)

{

try

{

if (letterNumberFromEnd != Processing.DatabaseUsers["Telegram." + e.Message.Chat.Id.ToString()]

.OutCity.Length - 1)

{

await \_botClient.SendTextMessageAsync(

chatId: e.Message.Chat,

text: $"Город {e.Message.Text} не найден в базе данных, попробуйте ввести другой город на эту же букву.",

replyMarkup: new InlineKeyboardMarkup(

new List<List<InlineKeyboardButton>>

{

new List<InlineKeyboardButton>

{

InlineKeyboardButton.WithCallbackData("Попробовать на следующую букву",

"NextLetter")

},

})

);

}

else

{

await \_botClient.SendTextMessageAsync(

chatId: e.Message.Chat,

text: $"Город {e.Message.Text} не найден в базе данных, попробуйте ввести другой город на эту же букву."

);

}

}

catch (KeyNotFoundException)

{

await \_botClient.SendTextMessageAsync(

chatId: e.Message.Chat,

text: $"Город {e.Message.Text} не найден в базе данных, попробуйте ввести другой город на эту же букву."

);

}

catch (Exception exception)

{

var s = $"OutCityNull - e.Message.Text: {e.Message.Text} - e.Message.Chat.Id: {e.Message.Chat.Id} - {exception}";

Logging.ERROR(s);

}

}

else

{

try

{

if (wikiUrl != null)

{

await \_botClient.SendChatActionAsync(

chatId: e.Message.Chat.Id,

chatAction: ChatAction.UploadPhoto

);

await \_botClient.SendPhotoAsync(

chatId: e.Message.Chat.Id,

caption: $"<b>{outCity}</b>",

parseMode: ParseMode.Html,

photo: photoUrl,

replyToMessageId: e.Message.MessageId,

replyMarkup: new InlineKeyboardMarkup(

new List<List<InlineKeyboardButton>>

{

new List<InlineKeyboardButton>

{

InlineKeyboardButton.WithUrl("Википедия", wikiUrl),

InlineKeyboardButton.WithUrl("Google", googleUrl),

InlineKeyboardButton.WithUrl("Яндекс", yandexUrl)

},

new List<InlineKeyboardButton>

{

InlineKeyboardButton.WithCallbackData("Больше информации о городе",

$"More|{outCity}")

},

})

);

}

else

{

await \_botClient.SendChatActionAsync(

chatId: e.Message.Chat.Id,

chatAction: ChatAction.UploadPhoto

);

await \_botClient.SendPhotoAsync(

chatId: e.Message.Chat.Id,

caption: $"<b>{outCity}</b>",

parseMode: ParseMode.Html,

photo: photoUrl,

replyToMessageId: e.Message.MessageId,

replyMarkup: new InlineKeyboardMarkup(

new List<List<InlineKeyboardButton>>

{

new List<InlineKeyboardButton>

{

InlineKeyboardButton.WithUrl("Google", googleUrl),

InlineKeyboardButton.WithUrl("Яндекс", yandexUrl)

},

new List<InlineKeyboardButton>

{

InlineKeyboardButton.WithCallbackData("Больше информации о городе",

$"More|{outCity}")

},

})

);

}

}

catch (Telegram.Bot.Exceptions.ApiRequestException)

{

try

{

if (wikiUrl != null)

{

await \_botClient.SendTextMessageAsync(

chatId: e.Message.Chat,

text: $"<b>{outCity}</b>",

parseMode: ParseMode.Html,

replyToMessageId: e.Message.MessageId,

replyMarkup: new InlineKeyboardMarkup(

new List<List<InlineKeyboardButton>>

{

new List<InlineKeyboardButton>

{

InlineKeyboardButton.WithUrl("Википедия", wikiUrl),

InlineKeyboardButton.WithUrl("Google", googleUrl),

InlineKeyboardButton.WithUrl("Яндекс", yandexUrl)

},

new List<InlineKeyboardButton>

{

InlineKeyboardButton.WithCallbackData("Больше информации о городе",

$"More|{outCity}")

},

})

);

}

else

{

await \_botClient.SendTextMessageAsync(

chatId: e.Message.Chat,

text: $"<b>{outCity}</b>",

parseMode: ParseMode.Html,

replyToMessageId: e.Message.MessageId,

replyMarkup: new InlineKeyboardMarkup(

new List<List<InlineKeyboardButton>>

{

new List<InlineKeyboardButton>

{

InlineKeyboardButton.WithUrl("Google", googleUrl),

InlineKeyboardButton.WithUrl("Яндекс", yandexUrl)

},

new List<InlineKeyboardButton>

{

InlineKeyboardButton.WithCallbackData("Больше информации о городе",

$"More|{outCity}")

},

})

);

}

}

catch (Exception exception)

{

var s = $"WithoutPic - outCity: {outCity} - e.Message.Text: {e.Message.Text} - e.Message.Chat.Id: {e.Message.Chat.Id} - {exception}";

Logging.ERROR(s);

}

}

catch (Exception exception)

{

var s = $"WithPic - outCity: {outCity} - e.Message.Text: {e.Message.Text} - e.Message.Chat.Id: {e.Message.Chat.Id} - {exception}";

Logging.ERROR(s);

}

}

}

}

1. Руководство пользователя

6.1 Область применения

Чат-бот применяется в развлекательных и образовательных целях, можно использовать для образования в развлекательной манере.

6.2 Краткое описание возможностей

Чат-бот способен вести с пользователем игру в Города (Города́ — игра, в которой каждый участник в свою очередь называет реально существующий город России, название которого начинается на ту букву, которой оканчивается название города предыдущего участника), помимо этого присутствует возможность у пользователя получить дополнительную информацию о городе или его местоположению.

6.3 Уровень подготовки пользователя

Использовать чат-бота может любой человек, который умеет читать и знает несколько российских городов (нужно для интересности игры).

6.4 Рекомендации по освоению

Пользователю достаточно уметь пользоваться мессенджером Telegram и иметь базовые знания городов России (для того, чтобы был интерес к игре).

1. Проектирование

Ниже приведено визуальное представление работы программы и взаимодействия различных ее компонентов:

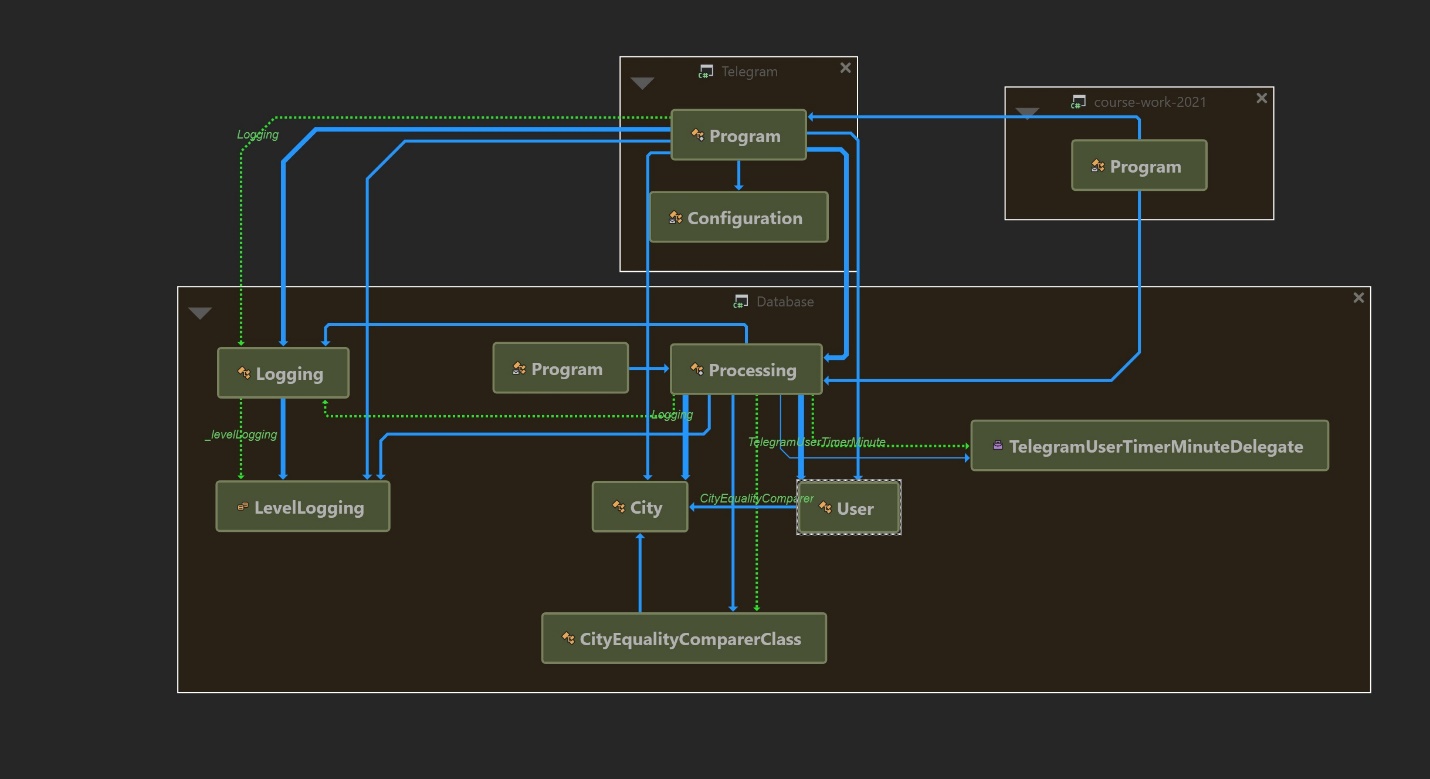


Рисунок 1 - Визуализация работы программы

1. Тестирование

В данной части отчета производится тестирование части работы, в которой производится подключение встроенных кнопок, которые дают пользователю возможность получить дополнительную информацию о данном городе. Тестирование производится посредством имитации реальной ситуации и использования бота наподобие реального пользователя. Ниже представлены скрин-шоты реакции программы(бота) на нажатие кнопок, тестирование которых производится в данном разделе.

Изображение выглядит как текст, электроника, дерево, снимок экрана

Автоматически созданное описание

Рисунок 2 - Пример работы чат-бота

Ниже представлена реакция программы на нажатие кнопки “Википедия”

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3 – Ссылка на википедию

Ниже представлена реакция программы на нажатие кнопки “Open”

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 4 – статься в википедии

Ниже представлена реакция программы на нажатие кнопки “Google”

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 5 - Google запрос

Ниже представлена реакция программы на нажатие кнопки “Яндекс”

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 6 - запрос Яндекс

Ниже представлена реакция программы на нажатие кнопки “Больше информации о городе”, которая осуществлена лишь для мобильного приложения, так как открывает мобильные карты и дает возможность построить маршрут до данного города.

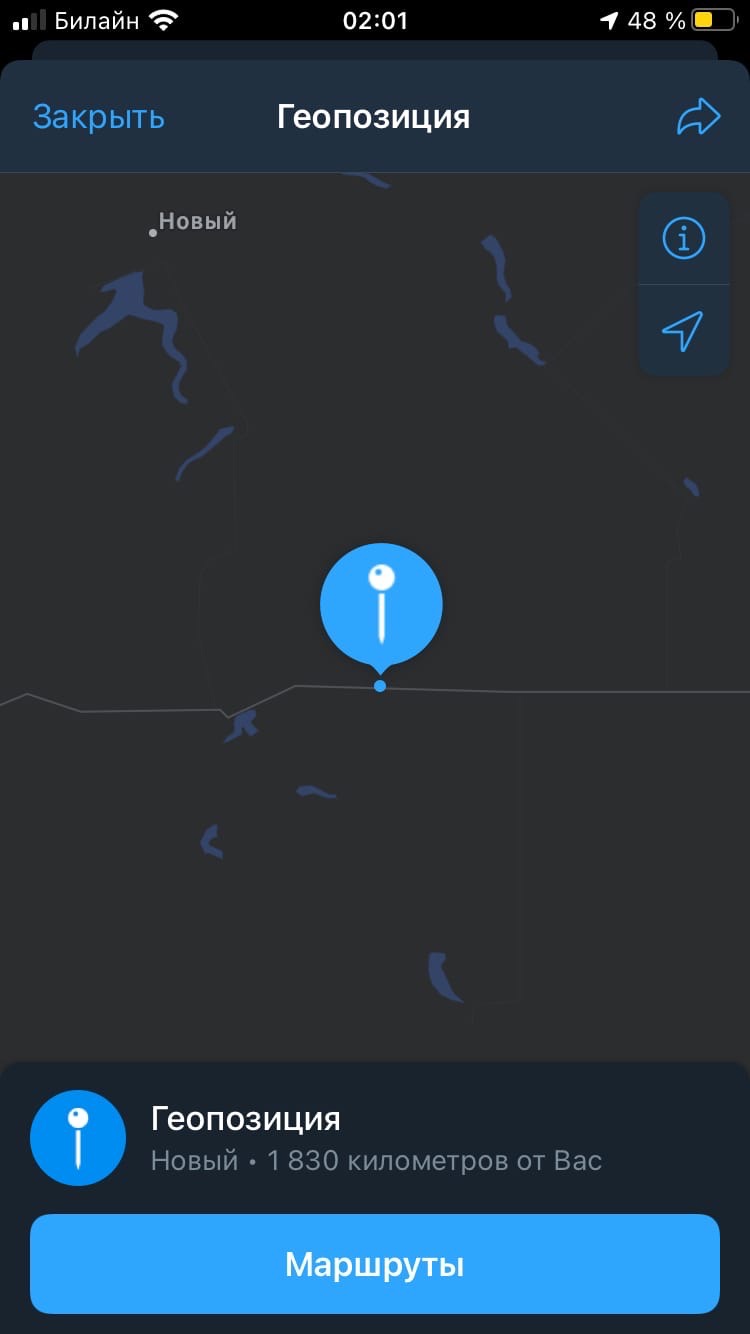


Рисунок 7 - кнопка "Больше информации о городе"

1. Заключение

В результате работы был создан чат-бот, который способен вести с пользователем игру в Города (Города́ — игра, в которой каждый участник в свою очередь называет реально существующий город России, название которого начинается на ту букву, которой оканчивается название города предыдущего участника), также бот способен предоставлять пользователю возможность получения справки из электронной библиотеки «Википедия» с подробной информацией о городе, фотографию города, которая опубликована в статье о городе в электронной библиотеке «Википедия», предоставлять ссылки в поисковые системы «Google» и «Яндекс» по запросу данного города, предоставлять пользователю геолокацию города с местоположением на карте (не поддерживается в веб-версии) и возможностью построить маршрут до данного города.  
Вся система работает согласно техническому заданию, все задуманные возможности реализованы.

1. Список используемых источников

**.NET Core в действии: пишем бота для telegram.** [В Интернете]. - 2 Апреля 2018 г.. - https://gosha20777.github.io/code/2018/04/02/dotnetcore-bot/.

**Aсинхронное программирование** [В Интернете]. - 17 Октября 2018 г.. - https://metanit.com/sharp/tutorial/13.3.php.

**nuget.org** [В Интернете]. - 2021 г.. - https://www.nuget.org/packages/LumenWorks.Framework.IO/.

**System.Linq Пространство имен** [В Интернете]. - 2021 г.. - https://docs.microsoft.com/ru-ru/dotnet/api/system.linq?view=net-5.0.

**Telegram bot API - разбор документации с примерами** [В Интернете]. - 2018 г.. - https://infostart.ru/1c/articles/1217332/.

**Атрибуты в .NET** [В Интернете]. - 23 Октября 2018 г.. - https://metanit.com/sharp/tutorial/14.4.php.

**Модули подключения для баз данных Microsoft SQL** [В Интернете]. - 2020 г.. - 2021 г.. - https://docs.microsoft.com/ru-ru/sql/connect/sql-connection-libraries?view=sql-server-ver15.

**Пишем бота telegram на C#** [В Интернете]. - 18 Марта 2017 г.. - http://aftamat4ik.ru/pishem-bota-telegram-na-c/.

**Справочник по Bot API** [В Интернете]. - 2016 г.. - https://tlgrm.ru/docs/bots/api#authorizing-your-bot.

**Что такое "lumenworks.framework.io.dll"?** [В Интернете]. - 2019 г.. - http://systemexplorer.net/ru/file-database/file/lumenworks.framework.io-dll.