Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение
высшего образования «Уральский федеральный университет
имени первого Президента России Б.Н.Ельцина»

Институт радиоэлектроники и информационных технологий – РТФ
Центр ускоренного обучения

НАПИСАНИЕ БОТА ДЛЯ СОЦИАЛЬНОЙ СЕТИ TELEGRAM НА ЯП JAVA

КУРСОВОЙ ПРОЕКТ
по дисциплине «Программирование»

Пояснительная записка
09.03.03 58.29.21 002 ПЗ

Ст. преподаватель: Н.А. Архипов
Нормоконтролёр: Н.А. Архипов
Студент группы РИЗ-100028у: И.С. Арсентьев

Екатеринбург 2021

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В 21 веке одним из самых ценных ресурсов в жизни человека является информация. Ведь именно с помощью информации человечество приобрело тот облик, которым обладает сейчас, уровень развития технологий за последнее столетие настолько шагнул вперед, что позволяет судить о громадном рывке в плане развития науки и технологий в целом. Таким образом, можно сказать, что в настоящий момент информация является основным двигателем прогресса и эволюции. Именно поэтому высокую ценность обретают источники, которые приносят определенную полезность людям, извлекающим информацию оттуда. Такие источники называются медиапродуктами. Одними из самых популярных агрегаторов медиапродуктов являются социальные сети в глобальной сети, которые на протяжении последних тридцати лет очень стремительно развивались.

С появлением социальных сетей и мессенджеров, таких как WhatsApp, Viber, Telegram, среда информационных технологий стала более доступной для абсолютно любого пользователя. Разнообразие получаемой информации из социальных сетей может быть настроена по желанию самого пользователя, на любой вкус и стремление для получения информации.

Telegram является одной из ведущих площадок медиа пространства. Важно отметить, что Telegram – это один из самых популярных мессенджеров на территории России и стран СНГ, является российской разработкой от создателя социальной сети ВКонтакте Павла Дурова. Изначально, мессенджер Telegram был создан с целью обмена сообщениями между пользователями. С течением времени, функционал мессенджера претерпел изменения и улучшения, а именно собеседником любого пользователя может стать автоматическая программа - робот, иными словами бот.

В Telegram существует три вида медиасервисов: чаты, каналы и боты. Телеграм-чаты создаются для обмена сообщениями между всеми участниками беседы. Телеграм-канал подразумевает отправку информации от одного лица, который является администратором. Бот – это собеседник в мессенджере, робот, отвечающий на запросы автоматически. Он заранее запрограммирован выдавать пользователю определенную информацию.

Актуальность исследования данной курсовой работы обосновывается необходимостью создания чат-бота для предоставления информации пользователю посредством ответа в сообщении в социальной сети Telegram.

В качестве языка программирования для создания данного сервиса используется мультипарадигмальный высокоуровневый язык программирования Java.

Объектом исследования в курсовой работе является процесс формирования запроса информации, поиск необходимой информации в Интернете, предоставление её пользователю по запросу чат-ботом.

Предмет исследования представляет собой программа на языке программирования Java, содержащая в себе компоненты для полноценной работы чат-бота, предоставляющего актуальную информацию пользователю в виде сообщения.

Задачей исследования в курсовой работе является создание автоматизированного сервиса для предоставления пользователю актуальных данных о погоде в заданной географической точке по названию места на карте с помощью парсинга данных веб-сайта, имеющем в своём содержании данную информацию.

После выполнения задачи, разработанный проект можно разместить на веб-сервисе для дальнейшего использования.

1 Постановка задачи

1.1 Регистрация бота в социальной сети Telegram

В социальной сети Telegram любой пользователь имеет аккаунт, будь то человек (реальный пользователь) или автоматизированная система отправки оповещений. Соответственно, для чат-бота необходима регистрация на сервисе Telegram. Данная процедура выполняется с помощью бота BotFather, который в свою очередь, позволяет получить токен для использования ботами в пространстве социальной сети Telegram.

Регистрация проходит очень просто и может быть выполнена за несколько минут пользователем. Для этого необходимо иметь на руках следующую информацию: имя бота и имя пользователя. Диалог с ботом BotFather представлен на рисунке 1 ниже:

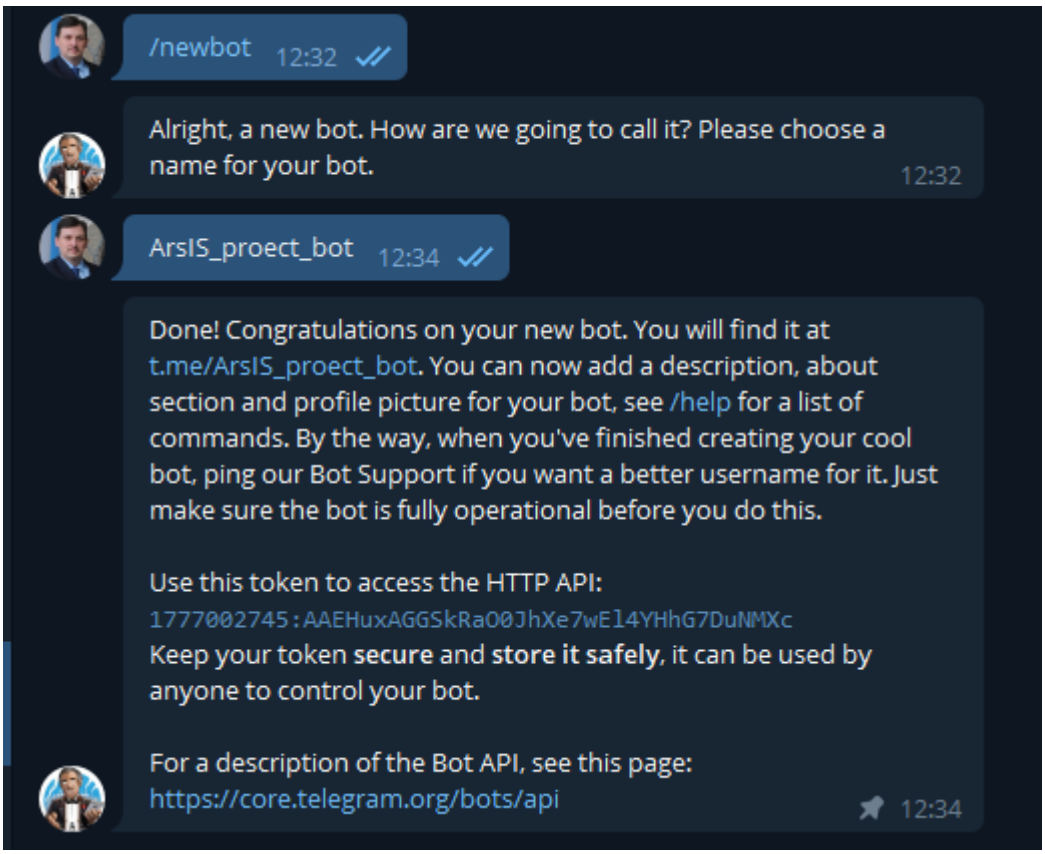

Рисунок 1 – Регистрация бота на сервисе @BotFather.

После регистрации бота необходимо непосредственное создание классов в среде разработки.

Для написания классов создаваемого чат-бота используем среду разработки Intellij IDEA community Edition 2020.3. На ПК установлена Java JDK 16 (x64). Переменная Path прописана корректно в папку с JDK на жёстком диске, что позволяет запускать написанные классы на виртуальной машине Java в ПК.

Одной из актуальных тем для сервиса выбрано предоставлении данных о погоде по заданному названию населённого пункта, данные будут браться с веб-сервиса https://openweathermap.org, который позволяет использовать API с данными о погоде.

Общую концепцию работы бота можно представить в виде блок-схемы (рисунок 2):

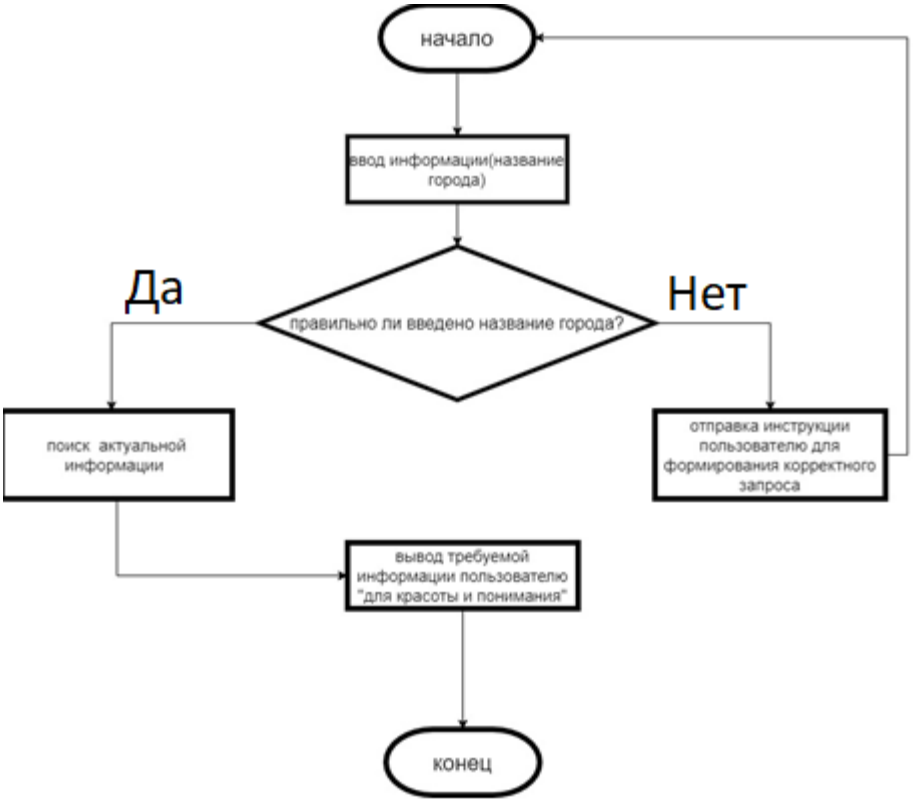

Рисунок 2 – Работа чат-бота в виде блок-схемы.

Таким образом, можно сказать, что бот будет выводить запрашиваемую информацию только в случае корректного ввода пользователем начальных данных, а именно название города или посёлка.

Использование бота поможет оперативно выяснять значения погоды для любого места на карте, информация будет предоставляться в онлайн режиме, при обращении к одному и тому же городу значения показателей будут меняться (а могут и оставаться неизменными, для уточнения деталей можно обращаться к другим сервисам в интернете).

Задачу по написанию программного продукта можно разделить на несколько этапов, от момента возникновения идеи до развёртывания на сервисе. Момент возможной модернизации после выпуска в первоначальный проект не включён, действия по обновлению сервиса возможны при создании отдельной подзадачи. Проект планируется для частного применения заинтересованному кругу лиц.

Более детально режим разработки данного проекта можно представить в виде таблицы:

Таблица 1. Общий алгоритм реализации проекта.

Этап	Выполняемые действия
1	Изучение материала, изучение принципов работы ботов с использованием работы API.
2	Регистрация бота на сервисе Telegram, выбор среды разработки на языке Java.
3	Создание блок-схемы работы будущего приложения, определение необходимых компонентов.
4	Написание необходимых классов в среде программирования.
5	Отладка и опытное использование на ПК запущенного бота.
6	Размещение бота на открытой площадке для использования без ПК.

2 Создание программного продукта

2.1 Написание классов бота в среде разработки Intellij IDEA.

Основным рабочим классом бота является класс Bot.java, который содержит в себе методы, которыми описываются запрос на веб-сервис, вывод информации в чат, кнопки клавиатуры для диалога с ботом, а также метод, возвращающий имя пользователя и метод, содержащий токен, выданный @BotFather при регистрации бота.

Для создания приложения необходимо подключить фрейворк Maven, который поможет нам автоматически создавать необходимые зависимости, поэтому при создании проекта выбираем соответствующую конфигурацию. Таким образом необходимые библиотеки подтянутся в проект, а в папке с проектом сформируется файл pom.xml, отображающий зависимости, работающие в проекте. Далее можно создавать основные классы.

Также для корректной работы сервиса необходим импорт библиотек, позволяющих нашему боту понимать информацию с серверов Telegram:

```
import org.telegram.telegrambots.ApiContextInitializer;
import org.telegram.telegrambots.TelegramBotsApi;
import org.telegram.telegrambots.api.methods.send.SendMessage;
import org.telegram.telegrambots.api.objects.Message;
import org.telegram.telegrambots.api.objects.Update;
import
org.telegram.telegrambots.api.objects.replykeyboard.ReplyKeyboar
dMarkup;
import
org.telegram.telegrambots.api.objects.replykeyboard.buttons.Keyb
oardButton;
import
org.telegram.telegrambots.api.objects.replykeyboard.buttons.Keyb
oardRow;
import org.telegram.telegrambots.bots.TelegramLongPollingBot;
import org.telegram.telegrambots.exceptions.TelegramApiException;
import
org.telegram.telegrambots.exceptions.TelegramApiRequestException
;
```

Также необходимы библиотеки для работы с массивом данных, получаемых с API веб-сервиса:

```
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
```

Итак, после импорта необходимых библиотек в основной класс Bot.java, необходимо инициализировать объект, с которым будет работать бот, а именно ApiContextInitializer.init() и создать объект TelegramBotsApi().

Далее в теле основного класса требуется определить методы, работающие в классе, т.е. метод для распознавания сообщения, отправленного боту, а также работу кнопок клавиатуры, размещённых в окне диалога с ботом, у нас их будет две: «Инструкция», которая позволит ориентировать пользователя по вводу и «Настройки», которая в дальнейшем может быть использована для работы с методом настроек в боте.

В методе public void onUpdateReceived будет формироваться объект сообщения, отправляемого пользователю ботом, в котором будет формироваться запрашиваемая информация о состоянии погоды, а также сообщения от бота в случае нажатия кнопок клавиатуры, класс Weather будет описан ниже и будет содержать информацию, какую назначим ему мы для вывода.

Вторым основным классом в программе будет Weather.java, который в себе содержит запрос на веб-ресурс. Для корректной работы также необходим импорт библиотек:

```
import org.json.JSONArray;
import org.json.JSONObject;
import java.io.IOException;
import java.io.InputStream;
import java.net.URL;
import java.util.Scanner;
public class Weather {
    public static String getWeather(String message, Model model)
throws IOException {// "извлечение информации" с web-страницы
ресурса
        URL                url                =                new
URL("http://api.openweathermap.org/data/2.5/weather?q=" + message
+ "&units=metric&applied=6fff53a641b9b9a799cfd6b079f5cd4e");
        //запрос по городу, который мы присвоили message
        Scanner      in      =      new      Scanner((InputStream)
url.getContent());//запуск объекта Scanner
```

```
        StringBuilder result = new StringBuilder();//создание переменной для вывода информации
        while (in.hasNext()) {// пробегаем циклом по запрашиваемым данным
            // формируем строки для вывода информации
        }
        JSONObject object = new JSONObject("result");
        model.setName(object.getString("name"));//присваивание значений,"извлечённых" с web-страницы ресурса
        JSONObject main = object.getJSONObject("main");
        model.setTemp(main.getDouble("temp"));
        model.setHumidity(main.getDouble("humidity"));
        JSONArray getArray = object.getJSONArray("weather");//формирование массива данных и проход по нему
        for (int i = 0; i < getArray.length(); i++) {
            JSONObject obj = getArray.getJSONObject(i);//объект, вытаскиваемый из массива данных с web-страницы
            model.setIcon((String) obj.get("icon"));//иконка для отображения статуса погоды
            model.setMain((String) obj.get("main"));//описание погоды    }
//вывод сообщения для пользователя, для "красоты и понимания" с указанием интересующей информации
        return "Place: " + model.getName() + "\n" +
                "Temperature: " + model.getTemp() + "C" + "\n" +
                "Humidity:" + model.getHumidity() + "%" + "\n" +
                "Weather: " + model.getMain() + "\n" +
                "http://openweathermap.org/img/w/" + model.getIcon() + ".png";
    }}
```

Очевидно, что в данном классе присутствует переменная величина, которая получает значения с веб-сервиса и формирует массив данных, которые необходимо преобразовать в форму, удобную для вывода в сообщении пользователю, «для красоты и понимания».

Поэтому необходимо создание объекта, который заполнится необходимыми данными, таким объектом является JSONObject object.

При обращении бота к сайту с запросом погоды по введённому названию места на карте, запрос выглядит следующим образом:

http://api.openweathermap.org/data/2.5/weather?q=" + message + "&units=metric&applied=6fff53a641b9b9a799cfd6b079f5cd4e

В данном запросе в качестве объекта message будет введённое пользователем название города, в конце запроса содержится ключ-токен, который мы получаем в процессе регистрации на веб-ресурсе, откуда берём данные.

Так как мы используем для получения информации ресурс https://www.openweathermap.org/, то для получения токена необходима регистрация, в процессе регистрации мы получаем свой токен, прикрепляемый к запросу. Итак, бот, отправляя запрос на сервер заданного веб-узла получает массив информации, содержащий значения погодных условий, таких как температура, влажность, наличие осадков, освещённость и т. д. Для «красоты и понимания» нам не требуется выводить всё, что нам ответит сервер, а ответ от него выглядит формально таким образом (рисунок 3): массив, заполненный значениями метрик, которые собирает сайт с доступных ему ресурсов. Таким образом, необходим метод, который будет приводить в порядок, и после сортировки выведет корректное сообщение для пользователя.

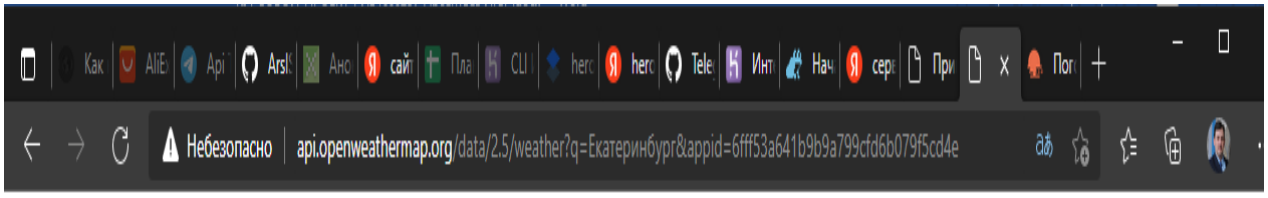

Рисунок 3 – Ответ сайта https://www.openweathermap.org по запросу погоды в городе Екатеринбург

Для этого нужно распарсить полученный с сервера запрос с целью создания нового объекта Model.java, который мы можем настроить должным образом для вывода нужной информации. Для вывода необходимой информации необходимо инициализировать переменные, которым мы будем задавать значения в соответствии с получаемыми данными:

место, температура, влажность, наличие/отсутствие осадков. Создаваемый класс примет вид:

```
public class Model{//класс с моделью сообщения, выдаваемого роботом
    private String name;//название населённого пункта
    private Double temp;//значение температуры воздуха
    private Double humidity;//значение влажности воздуха
    private String icon;//значение картинки,отображающей статус погоды
    private String main;//значение общего статуса погоды
    //сгенерированные методы для "вытаскивания данных"
    public String getName() { return name; }
    public void setName(String name) {  this.name = name;  }
    public Double getTemp() { return temp; }
    public void setTemp(Double temp) { this.temp = temp; }
    public Double getHumidity() { return humidity;}
    public void setHumidity(Double humidity) { this.humidity = humidity; }
    public String getIcon() { return icon;}
    public void setIcon(String icon) {this.icon = icon;}
    public String getMain() {return main; }
    public void setMain(String main) { this.main = main; }
}
```

Таким образом, сгенерированные методы возвращают пользователю только те данные, которые описаны в данном классе. Если требуется отображение каких-либо других данных, тогда требуется ввести необходимые переменные и добавить их в класс Model.java. В итоге, в классе Weather.java фактом отработки всех действий формируется сообщение пользователю по его запросу (рисунок 4, 5).

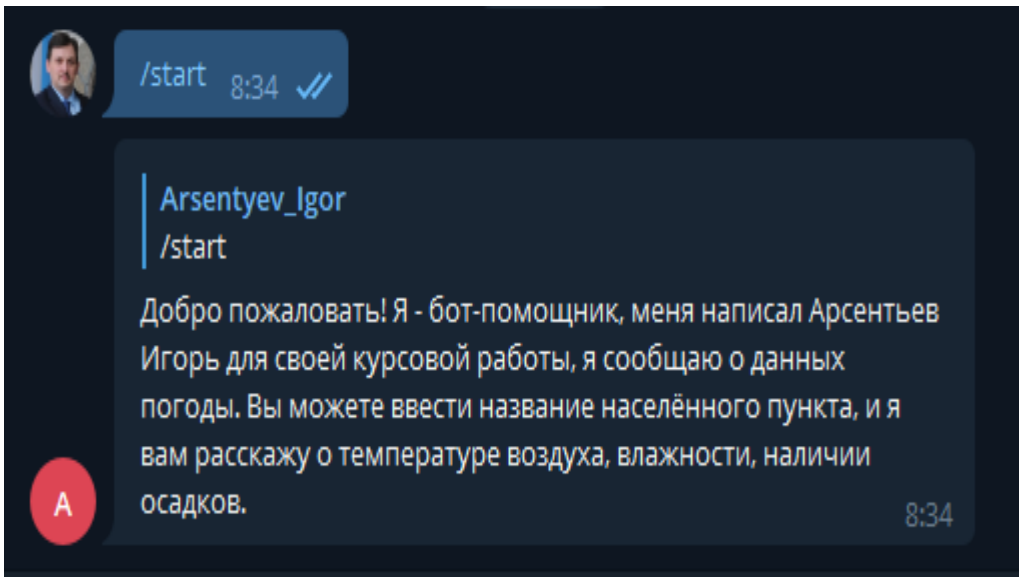

Рисунок 4 – Иллюстрация работы бота (на ПК)

Бот доступен с любых устройств, которые используют Telegram, как стационарный ПК, так и мобильное устройство (планшет, смартфон), для визуального отображения информации бот разные типы данных: вещественные числа, строки, изображение (рисунок 5):

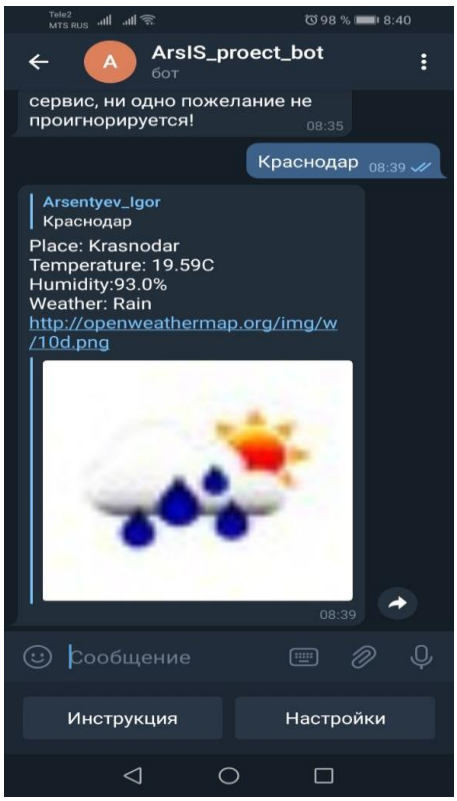

Рисунок 5 – Результат работы бота в отображении на смартфоне

Также, при нажатии на кнопки оформленной нами клавиатуры, бот выдаёт соответствующие сообщения (рисунок 6):

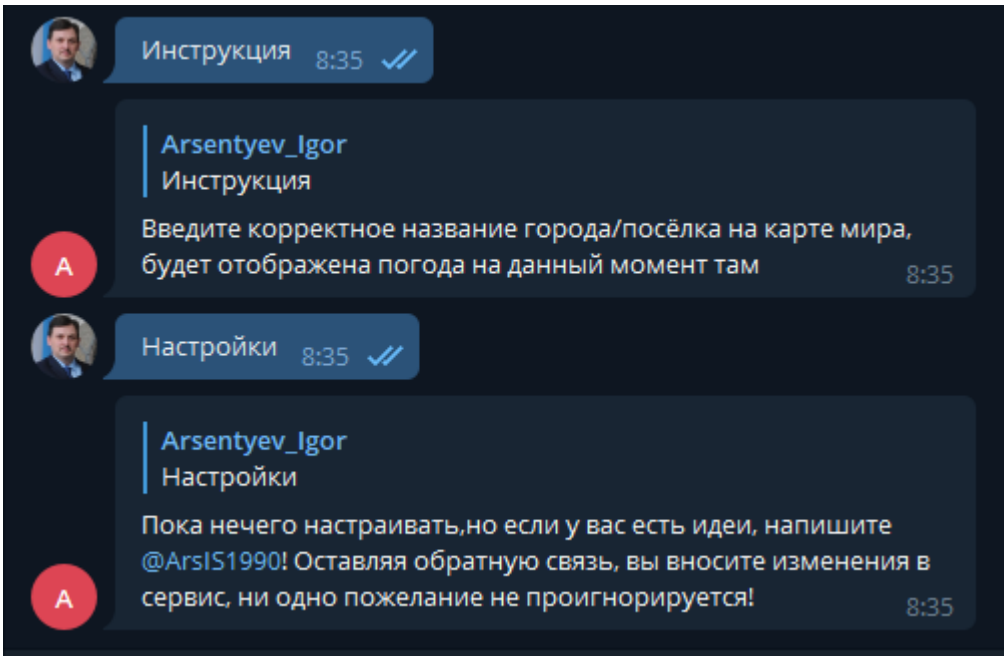

Рисунок 6 – Результат работы бота при нажатии на интерактивные кнопки в окне диалога

Когда бот будет работать в среде разработки, его можно отправлять на сервис публикации проектов.

3 Размещение готового продукта в открытом доступе.

3.1 Публикация проекта в интернет-пространстве.

Пока проект запущен только на ПК разработчика, этот локальный проект. Возникает вопрос: каким образом можно получить доступ к разработанному сервису в формате 24/7?

На этот вопрос есть ответ, и он прост: проекту надо дать жизнь в открытом интернет-пространстве. Существует множество площадок для публикации подобных ботов, которые позволяют, размещая в интернете проект, иметь доступ к нему в любое время с любого устройства.

Можно привести список сервисов и способы размещения, большинство сервисов в сети подразумевают под собой аренду оборудования, а именно сервера с настраиваемыми параметрами.

Если мы имеем для запуска проект, сосредотачивающий работу с сервисами с участием большого количества пользователей, необходимо подсчитать параметры оборудования, которое обработает с необходимой нам скоростью запросы пользователей. Множество коммерческих площадок позволяют вручную настроить параметры виртуальной машины, которую вы можете взять в аренду и разместить на нём свою программу, в данном случае бота.

Также имеются альтернативные условно бесплатные сервисы, например heroku.com, на которую можно загрузить свой проект прямо с площадки GitHub, инициировать запуск приложения и пользоваться без привлечения физического оборудования.

Процесс размещения бота на хостинге heroku.com несложен – на сайте представлена подробная инструкция по размещению проекта из GitHub, после регистрации на сайте можно создать приложение и произвести запуск.

При размещении проекта на реальном сервере необходимо учесть тот факт, что на сервере присутствовали компоненты, необходимые для запуска. В противном случае перед запуском необходимо вручную их устанавливать.

Java – не единственный язык, на котором можно создавать Telegram-ботов, подобных ботов пишут также на Python, JavaScript, C#, C++ и др.

3.2 Альтернативные языки для написания подобного бота.

Как и сказано выше, на выбор можно взять любой язык программирования для написания подобного бота. На данный момент особо популярным языком для такого проекта многие признают Python, их выбор основывается на том, что у Python более низкий порог для вхождения в процесс программирования. С современным развитием информатики, даже многие школьники разрабатывают свои первые программы именно на Python. Так как работа бота завязывается на фрейворках, то при написании данной

программы нужно учитывать наличие необходимых компонентов в системе для корректной работы приложения. Поскольку для размещения ПО в интернете при выборе платформы многие выбирают дистрибутивы Linux, необходимые библиотеки для Python присутствуют в системе.

Другим конкурентом для разработки данного бота может послужить JavaScript. «Младший брат» Java тоже может являться отличным решением для написания бота и дальнейшем его использовании, при наличии навыка у пользователя в написании скриптов на данном языке программирования, бот также будет стабильно работать с API указанных веб-ресурсов.

Вообще выбор языка программирования для написания подобного бота (как и многих других) весьма разнообразен: всё зависит здесь только от самого программиста, ведь лучше всего можно описать логику работы бота на языке, который вы знаете, а ерунда может получиться в любом языке, поэтому создание ботов стоит проводить только в том языке, синтаксис и принципы работы логики объектов вам хорошо известны.

Возникает вопрос: а доступно ли создание такого сервиса как чат-бот для человека, который не имеет понятия о том, что такое программирование или знания его в этой сфере минимальны? Возможно ли создать бота без привлечения таких инструментов как среда разработки, компилятор и виртуальная машина на физическом ПК?

В таком случае можно сказать, что в интернете существуют сервисы для создания ботов для социальных сетей без навыков программирования.

3.3 Способы создания чат-ботов без опыта программирования.

Тут снова стоит упомянуть о назначении такого сервиса, как чат-бот. Если нам необходимо создать бота с минимальным набором операций, то для его создания можно найти альтернативный метод, а именно использование конструктора.

В широком смысле, человечество начало интенсивно пользоваться веб-ресурсами с начала 2000-х годов, примерно тогда же начали появляться веб-конструкторы для создания страниц интернета, сайтов. Аналогично, с популяризацией социальных сетей и применения чат-ботов, некоторые платформы предлагают услуги своих конструкторов для написания бота для социальной сети с выбором необходимых функций. Соответственно, услуги данных сервисов условно бесплатны, это вызвано набором необходимых функций при создании сервиса.

К примеру, https://chatfuel.com/, анонсированный Яндексом, позволяет создать самостоятельно бота в социальной сети Facebook. Путём манипуляций при создании страницы в конечном итоге мы получаем веб-страницу создаваемого пользователя (рисунок 7):

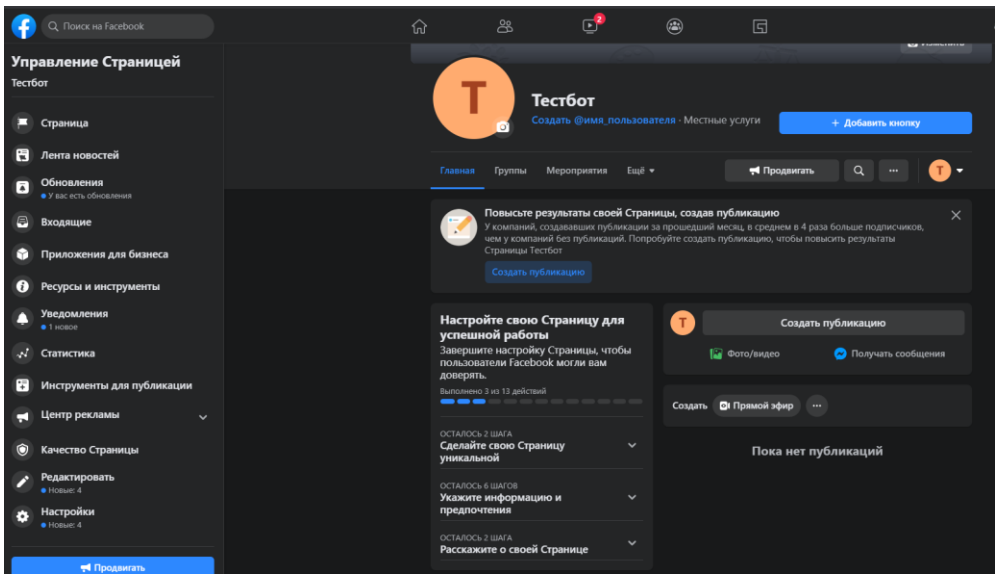

Рисунок 7 – Бот в социальной сети Facebook.

В настраиваемых функциях можно выставить настройку оповещений в другие социальные сети о событиях или письмах, поступивших боту от других пользователей. Более подробно информация о сервисах по созданию ботов для Telegram приведена в таблице 2(приложение).

Таким образом, можно сказать, что для создания Telegram-бота не нужно иметь специфических познаний в области программирования, системы искусственного интеллекта на выше предложенных площадках подберут под ваши вкусы и запросы все необходимые компоненты, вопрос оплаты сервисов опять-таки варьируется от используемых функций.

3.4 Востребованность на рынке труда специальности, связанной с написанием программ чат-ботов для социальной сети.

Ввиду того, что написание чат-ботов не является специфической задачей, а является задачей прикладного характера, то выполнять работы по созданию данной программы может junior- разработчик.

Выбор уровня разработки обусловлен тем, что даже для написания чат-бота необходимы базовые знания работы языка программирования и среды разработки, поэтому написание классов программы осилит начинающий программист. Наличие справочного материала на просторах интернета позволяют самостоятельно написать и наладить работу сервиса, связанного с социальной сетью.

Уровень оплаты труда начинающего разработчика в России изображен на гистограмме ниже (рисунок 8):

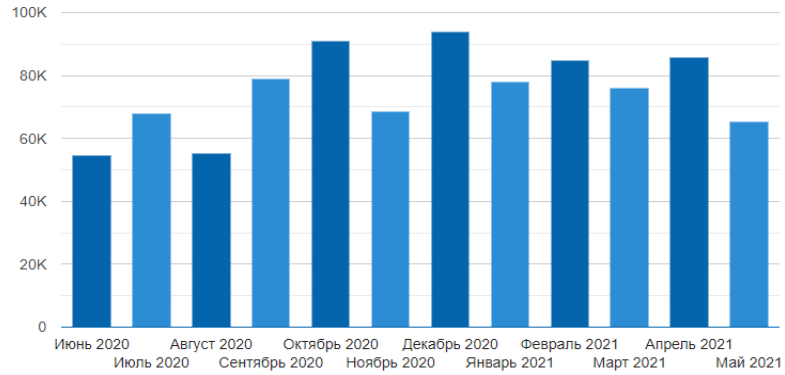

Рисунок 8 – Уровень оплаты труда junior Java developer в РФ (2021 г.)

ЗАКЛЮЧЕНИЕ

Многообразие современных социальных сетей позволяет использовать такие программы как чат-боты для разнообразных задач, будь то информационная рассылка, консультация, опрос по вопросам качества и т.п.

В вопросах создания данного инструмента, как чат-бот необходимо отталкиваться от специфики предоставляемой услуги, ведь от количества необходимых функций зависит наличие тех или иных классов ив тексте программы.

Выбор языка программирования для создания сложного бота зависит от навыков разработчика, под этим понимается степень осведомлённости о работе механизмов бота при запуске его в глобальной сети.

Размещение готового бота можно производить на различных площадках: от свободно используемых в сети, до коммерческих сервисов с набором функций по обслуживанию программного продукта.

На языке программирования Java в ходе курсовой работы создан бот для социальной сети Telegram, который предназначен для сообщения пользователю по запросу сведений о погоде по заданному названию места на карте. Алгоритм реализован на механизме парсинга данных, получаемых от веб-сервиса, и выводится на экран пользователю в виде текстового сообщения с данными, также реализована возможность передачи статусной картинки погоды в сообщении. В активном окне диалога присутствуют две интерактивные кнопки, при нажатии на которые, пользователю приходят автоматически сформированные сообщения роботом. Включена возможность обработки исключений- в случае некорректного ввода названия пользователем, бот предупреждает в сообщении о некорректном вводе.

Проект, находящийся в репозитории разработчика (Арсентьев И.С.), является открытым для доступа и может быть использован для дальнейшей доработки с целью повышения качества предоставляемого сервиса. Данный проект был разработан по заданию курсовой работы, выполнение необходимых функций бота проверена путём запуска в среде разработки, данный бот может быть опубликован на ресурсах интернета на необходимых площадках.

В конце 3 главы приведены способы создания альтернативных чат-ботов в социальной сети, в приложении приведена таблица примера сервисов, которые были проанализированы на предмет создания на их платформе ботов без использования среды разработки пользователем самостоятельно.

В приложении к курсовой работе находятся тексты основных классов бота, написанные на языке программирования Java.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1) Аванесян Н. Л., Telegram, как пример мессенджера: возможности и перспективы развития. [Электронный ресурс] / Н. Л. Аванесян // Научный потенциал XXI века. – 2017. – Режим доступа: https://elibrary.ru/download/elibrary_29653726_34734017.pdf (дата обращения 25.05.2021).
2) Архаков, Д. NodeJS: Делаем кнопки в Telegram API (inline-keyboards) [Электронный ресурс] / Д. Архаков // Блог о программировании. – 2016. – Режим доступа: https://archakov.im/post/nodejs-make-buttons-on-telegram-api.html (дата обращения 30.05.2021).
3) 3.Васильев, А.Н. Программирование на Java для начинающих / А.Н. Васильев. - М.: Эксмо, 2014. - 416 с.
4) Давыдов, С. IntelliJ IDEA. Профессиональное программирование на Java / С. Давыдов. - СПб.: BHV, 2005. - 800 с.
5) Иванов А. Д., Чат-бот в Telegram и ВКонтакте, как новый канал распространения новостей. [Электронный ресурс] / А. Д. Иванов // Волжский университет имени В.Н. Татищева. – 2016. – №3 – с. 126-132. – Режим доступа: https://elibrary.ru/download/elibrary_26673675_34058358.pdf (дата обращения 15.03.2021).
6) Козлов А. А., Телеграм-бот как простой и удобный способ получения информации [Электронный ресурс] / А. А. Козлов, А. В. Батищев //Территория науки. – 2017. – №5. – с. 55-64. – Режим доступа: https://cyberleninka.ru/article/v/telegram-bot-kak-prostoy-i-udobnyy-sposob-polucheniya-informatsii (дата обращения 10.03.2021).
7) Матвеева Н. Ю., Технологии создания и применения чат-ботов [Электронный ресурс] / Н. Ю. Матвеева, А. В Золотарюк . // Научные записки молодых исследователей. – 2018. – №1. – с. 28-30. – Режим доступа: https://cyberleninka.ru/article/v/tehnologii-sozdaniya-i-primeneniya-chat-botov (дата обращения 20.04.2021).
8) Машнин, Т.С. Web-сервисы Java. Профессиональное программирование / Т.С. Машнин. - СПб.: BHV, 2012. - 560 с.
9) Официальный сайт Heroku [Электронный ресурс]: Облачная PaaS-платформа. – Режим доступа: https://www.heroku.com/ (дата обращения 10.06.2021).
10) Официальный сайт Telegtam [Электронный ресурс]: API – Режим доступа: https://core.telegram.org/api (дата обращения 1.03.2021).

11) Герберт, Шилдт Java 8. Руководство для начинающих / Шилдт Герберт. - М.: Диалектика / Вильямс, 2015. - 899 с.
12) Шмыров, В. Названы любимые мессенджеры россиян [Электронный ресурс] / В. Шмыров // Издание о высоких технологиях. – 2018 – Режим доступа: http://www.cnews.ru/news/top/2018-02 28_whatsapp_stal_samym_populyarnym_messendzheram_v (дата обращения 3.03.2021)

Приложение А

(справочное)

Текст программ(основных классов)

Основной класс бота Bot.java

```
import org.telegram.telegrambots.ApiContextInitializer;
import org.telegram.telegrambots.TelegramBotsApi;
import org.telegram.telegrambots.api.methods.send.SendMessage;
import org.telegram.telegrambots.api.objects.Message;
import org.telegram.telegrambots.api.objects.Update;
import
org.telegram.telegrambots.api.objects.replykeyboard.ReplyKeyboar
dMarkup;
import
org.telegram.telegrambots.api.objects.replykeyboard.buttons.Keyb
oardButton;
import
org.telegram.telegrambots.api.objects.replykeyboard.buttons.Keyb
oardRow;
import org.telegram.telegrambots.bots.TelegramLongPollingBot;
import org.telegram.telegrambots.exceptions.TelegramApiException;
import
org.telegram.telegrambots.exceptions.TelegramApiRequestException
;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
public class Bot extends TelegramLongPollingBot {//наследование
класса от TelegramLongPollingBot
    public static void main(String[] args) {// точка входа в проект
        ApiContextInitializer.init();//инициализация Api
        TelegramBotsApi telegramBotsApi = new
TelegramBotsApi();//создание объекта TelegramApi
        try {telegramBotsApi.registerBot(new Bot());}
```

```
        catch (TelegramApiRequestException e) {// вывод на экран в случае,когда выбрасывается исключение
            e.printStackTrace();
        }
    }//переопределение методов
    public void sendMsg(Message message, String text) {//метод, описывающий действия робота в ответ на наши сообщения ему
        SendMessage sendMessage = new SendMessage();// инициализация сообщения,отправленного роботу
        sendMessage.enableMarkdown(true);// включение возможности разметки

sendMessage.setChatId(message.getChatId().toString());//какому чату ответить и на какое сообщение ответить боту
        sendMessage.setReplyToMessageId(message.getMessageId());
        sendMessage.setText(text);//текст,отправленный в метод
        try {
            setButtons(sendMessage);//отправка сообщения
            sendMessage(sendMessage);//устаревший, но работающий метод
        } catch (TelegramApiException e) {
            e.printStackTrace();
        }
    }
    public void onUpdateReceived(Update update) {// метод для приёма сообщений(обновление)
        Model model = new Model();//объект модели,отправляемой клиенту
        Message message = update.getMessage();//создание объекта message
        if (message != null && message.hasText()) {//проверка на корректность ввода сообщение(пустое или нет)
            switch (message.getText()) {//реакция бота на нажатие кнопок клиентом
case "/":
```

```
                    sendMsg(message, "Добро пожаловать! Я - бот-помощник, меня написал Арсентьев Игорь для своей курсовой работы, я сообщаю о данных погоды. Вы можете ввести название населённого пункта, и я вам расскажу о температуре воздуха, влажности, наличии осадков.");//сообщение-приветствие пользователю
                    break;
                case "/start":
                    sendMsg(message, "Добро пожаловать! Я - бот-помощник, меня написал Арсентьев Игорь для своей курсовой работы, я сообщаю о данных погоды. Вы можете ввести название населённого пункта, и я вам расскажу о температуре воздуха, влажности, наличии осадков.");//сообщение-приветствие пользователю
                    break;case "Инструкция"://  кнопка "Инструкция"
                    sendMsg(message, "Введите корректное название города/посёлка на карте мира, будет отображена погода на данный момент там");//сообщение пользователю, для "красоты и понимания"
                    break;
                case "Настройки"://кнопка "Настройки"
                    sendMsg(message, "Пока нечего настраивать, введите название города/посёлка на карте мира, мы работаем над новым функционалом1");
                    // сообщение на кнопке настроек
                    break;
                default://вывод информации с сайта по введённому населённому пункту,если таковой имеется
                    try { sendMsg(message, Weather.getWeather(message.getText(), model));//сообщение клиента и ответ ему с моделью
                    } catch (IOException e) {//сообщение в случае некорректного ввода
                        sendMsg(message, "Введите КОРРЕКТНОЕ название города/посёлка на карте мира!");//сообщение клиенту при ошибке ввода
                    }} } }
```

```
    public void setButtons(SendMessage sendMessage) {//метод,описывающий клавиатуру в чате
        ReplyKeyboardMarkup replyKeyboardMarkup = new ReplyKeyboardMarkup();//создание клавиатуры для "красоты и понимания"

sendMessage.setReplyMarkup(replyKeyboardMarkup);//разметка для клавиатуры

replyKeyboardMarkup.setSelective(true);//параметр,определяющий,кому выводить клавиатуру на экран-по умолчанию клиенту,который вошёл в чат

replyKeyboardMarkup.setResizeKeyboard(true);//автоматическая подгонка размера клавиатуры
        replyKeyboardMarkup.setOneTimeKeyboard(false);//скрывать или нет клавиатуру
        List<KeyboardRow> keyboardRowList = new ArrayList<>();
        KeyboardRow keyboardFirstRow = new KeyboardRow();
//создание кнопок клавиатуры
        keyboardFirstRow.add(new KeyboardButton("Инструкция"));
        keyboardFirstRow.add(new KeyboardButton("Настройки"));
//список кнопок клавиатуры
        keyboardRowList.add(keyboardFirstRow);

replyKeyboardMarkup.setKeyboard(keyboardRowList);//установка списка на клавиатуре
    }
    public String getBotUsername() {// метод для возврата имени,указанного при регистрации
        return "ArsIS_proect_bot";
    } public String getBotToken() {//введённый токен,выданный BotFather
        return "1777002745:AAEHuxAGGSkRaO0JhXe7wEl4YHhG7DuNMXc";
    }}
```

Основной класс Weather.java

```
import org.json.JSONArray;
import org.json.JSONObject;
import java.io.IOException;
import java.io.InputStream;
import java.net.URL;
import java.util.Scanner;
public class Weather {
    public static String getWeather(String message, Model model)
throws IOException {// "извлечение информации" с web-страницы
ресурса
        URL                url                =                new
URL("http://api.openweathermap.org/data/2.5/weather?q=" + message
+ "&units=metric&appid=6fff53a641b9b9a799cfd6b079f5cd4e");
        //запрос по городу,который мы присвоили message
        Scanner      in      =      new      Scanner((InputStream)
url.getContent());//запуск объекта Scanner
        String result = "";//создание переменной для вывода
информации
        while (in.hasNext()) {// пробегаем циклом по запрашиваемым
данным
            result += in.nextLine();//формируем строки для вывода
информации   }
        JSONObject object = new JSONObject(result);
        model.setName(object.getString("name"));//присваивание
значений,"извлечённых" с web-страницы ресурса
        JSONObject main = object.getJSONObject("main");
        model.setTemp(main.getDouble("temp"));
        model.setHumidity(main.getDouble("humidity"));
        JSONArray                  getArray                  =
object.getJSONArray("weather");//формирование массива данных и
проход по нему
        for (int i = 0; i < getArray.length(); i++) {
            JSONObject obj = getArray.getJSONObject(i);//объект,
вытаскиваемый из массива данных с web-страницы
```

```
            model.setIcon((String)  obj.get("icon"));//иконка  для
отображения статуса погоды
            model.setMain((String)    obj.get("main"));//описание
погоды
        }
//вывод сообщения для пользователя,для "красоты и понимания" с
указанием интересующей информации
        return "Place: " + model.getName() + "\n" +
                "Temperature: " + model.getTemp() + "C" + "\n" +
                "Humidity:" + model.getHumidity() + "%" + "\n" +
                "Weather: " + model.getMain() + "\n" +
                "http://openweathermap.org/img/w/"              +
model.getIcon() + ".png";
    }
}
```

Класс Model.java

```
public class Model {//класс с моделью сообщения, выдаваемого роботом
    private String name;//название населённого пункта
    private Double temp;//значение температуры воздуха
    private Double humidity;//значение влажности воздуха
    private String icon;//значение картинки,отображающей статус погоды
    private String main;//значение общего статуса погоды
    //сгенерированные методы для "вытаскивания данных"
    public String getName() { return name;}
    public void setName(String name) {this.name = name;}
    public Double getTemp() {return temp;}
    public void setTemp(Double temp) { this.temp = temp;}
    public Double getHumidity() { return humidity; }
    public void setHumidity(Double humidity) { this.humidity = humidity; }
    public String getIcon() {return icon;}
    public void setIcon(String icon) { this.icon = icon;}
    public String getMain() { return main;}
    public void setMain(String main) {
        this.main = main;
    }}
```

Сервисы создания ботов для социальных сетей без навыков программирования

Таблица 2. Список сервисов по созданию ботов для Telegram.

Наименование сервиса	Описание	Стоимость	Наличие мобильной платформы	Опыт использования
Manybot	Конструктор позволяет создавать меню, подменю, форму обратной связи, подключать RSS-ленты и делать рассылки по всем подписанным пользователям.	бесплатно (с рекламой, где создан бот). Поменять тариф или отключить рекламное сообщение нельзя.	есть	сервис идеально подходит для небольших проектов, но создавать меню более четырёх уровней через интерфейс бота становится сложно. Помимо этого, бот долго отвечает (или не отвечает совсем).

Botobot	узконаправленный сервис, который позволяет создать бот-магазин. Загрузка товаров происходит через Excel. Можно делать рассылки новостей магазина, отсутствует настройка меню и вообще сделать что-то кроме магазина (или как-то кастомизировать магазин) не представляется возможным.	бесплатный тариф (бессрочно, ограничение 20 товаров, рекламное сообщение сервиса). Минимальный платный тариф — 640 рублей в месяц.	есть	сильно ограниченный сервис в плане дополнительных возможностей и настроек. Надеялся на оплату непосредственно через Telegram, но такую функцию не обнаружил. Есть возможность на платном тарифе подключить CRM или настроить API.

Bottap	конструктор со стандартными возможностями: можно создать меню, формы обратной связи, интернет-магазин, онлайн-запись, делать простые рассылки.	бесплатный тариф (бессрочно, рекламное сообщение сервиса, нет интернет-магазина и онлайн-записи). Минимальный платный тариф — 390 рублей в месяц.	есть	сервис с приятным и вполне понятным личным кабинетом. Подходит для средних (по глубине вкладок) ботов. При создании больших ботов появились сложности (исходя из реализации просмотра команд в конструкторе). Присутствуют странности: у некоторых блоков нет кнопки «Удалить» (блок обратной связи), кнопки в меню нельзя расположить по желанию пользователя, нельзя изменить сообщение,

				отправляемое с клавиатуры, и прочее.
Chatforma	имеется вся необходимая функциональность: форма ввода, интернет-магазин, разные опросы, рассылки и всё то, что должно быть (не нашла только общение через бота). Имеется возможность создать теги для распознавания команд. Большое количество сервисов для интеграции.	триал-версия (15 дней с полной функциональностью). Минимальный платный тариф — 4000 рублей за два месяца, дальше по 2000 рублей в месяц	нет	большой сервис с активной поддержкой сообщества (видеоуроки, конкурсы и прочее). Содержит в себе необходимую функциональность для создания серьёзного бота. Простой и понятный личный кабинет. Нет чего-то особенного (разве что опросы), но есть всё необходимое.

Socialbot	умеет выводить информацию и отображать клавиатуру. Помимо этого, есть восемь модулей, в числе которых: вопросы, общение с пользователями, информация о новинках с сайтов Lostfilm и Seasonvar, простой заказ и гадание таро.	бесплатный тариф (бессрочно, нет рекламного сообщения сервиса, есть ограничение по модулям). Минимальный платный тариф — 500 рублей в месяц.	нет	слабый по возможностям и по удобству в работе сервис. Сервис не обновляется с 2016 года.

Botmother	платформа с множеством различных функций (если получится разобраться, как их использовать) и интеграциями. Длинная «палитра» блоков для конструктора, диалоги с пользователем, рассылки и не только.	бесплатный тариф (бессрочно, нет рекламного сообщения сервиса, из вкладок доступен только конструктор с ограничениями). Минимальный платный тариф — 1499 рублей в месяц.	нет	конструктор показался сложен в освоении из-за обилия функций и мессенджеров, которые можно подключить (что-то доступно для одного, но недоступно для другого).Интерфейс личного кабинета сильно напоминает Ai mylogic

Aimylogic	сервис по созданию всеми привычных «текстовых» ботов и непривычных ботов для обзвонов (для платного тарифа). Интересные, но сложные для освоения функции, такие как: «интенты» и «сущности». Подойдёт для создания ИИ и захвата мира.	бесплатный тариф (бессрочно, нет рекламного сообщения сервиса, есть ограничение по функциям и по количеству пользователей). Минимальный платный тариф — 5900 рублей в месяц.	нет	интерфейс конструктора выглядит похожим на Botmother (н о Aimylogic на много старше, а следовательно, можно предположить, кто на кого похож). В нём нет привычных функций: корзина, создание инлайн-клавиатуры, отправки файла, но можно создать бота, который будет угадывать то, о чём ему пишут.

Приложение Б

Скриншот с сайта Антиплагиат.

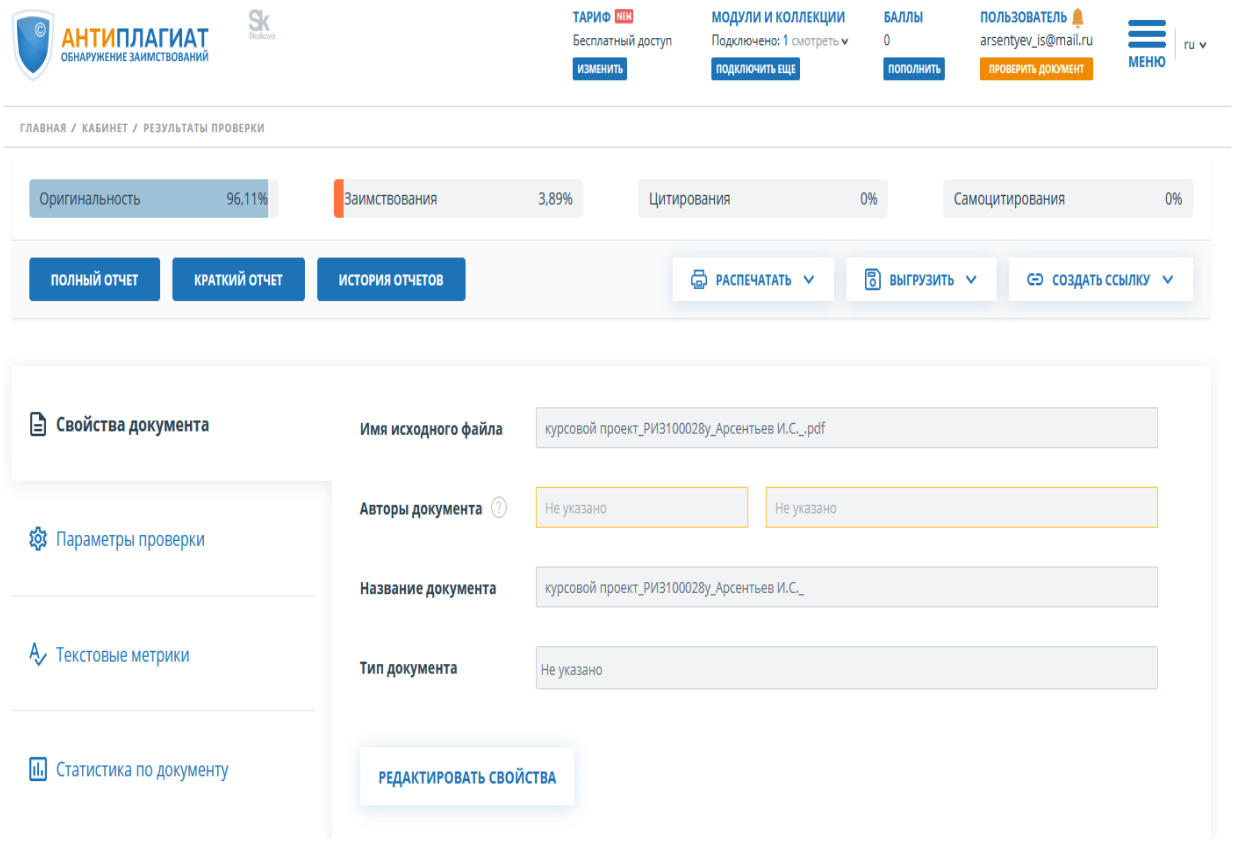

Рисунок 9 – Скрин с сайта Антиплагитат(https://www.antiplagiat.ru)