МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Белорусский национальный технический университет

Факультет Информационных технологий и робототехники

Кафедра «Программное обеспечения информационных систем и технологий»

КУРСОВОЙ ПРОЕКТ

По дисциплине

“Основы автоматизации конструирования”

Тема: “Новостной Telegram-бот”

Выполнил: студент группы 10702319

Наумов Антон Михайлович

Руководитель: доцент

Кункевич Дмитрий Петрович

Минск, 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Белорусский национальный технический университет

Факультет Информационных технологий и робототехники

Кафедра «Программное обеспечения информационных систем и технологий»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОМУ ПРОЕКТУ

По дисциплине

“Основы автоматизации конструирования”

Тема: “Новостной Telegram-бот”

Выполнил: студент группы 10702319

Наумов Антон Михайлович

Руководитель: доцент

Кункевич Дмитрий Петрович

Минск, 2022

**Оглавление**

Введение…………………………………………………………………….4

Обзор предметной области………………………………………………...5

Теоретические сведения об интернет-ботах……………………………...5

Назначение приложения…………………………………………………...6

Область применения и условия функционирования……………………..6

Обзор аналогов……………………………………………………………...7

Модель прецедентов………………………………………………………..9

Функциональная модель приложения…………………………………...12

Заключение………………………………………………………………...17

Литература…………………………………………………………………18

Приложение 1……………………………………………………………...19

**Введение**

В современном мире для общения, развлечений, работы и обучения

повсеместно используется Интернет. Интернет является не только

источником разнообразной и полезной для пользователей информации, но

также является основной формой виртуального общения. Благодаря

Интернету теперь не требуется постоянное личное присутствие, многие

вопросы решаются через средства виртуальной коммуникации. Во многом

этому поспособствовали системы мгновенного обмена сообщениями –

мессенджеры. Мессенджеры обеспечивают пользователей большим

количеством удобных способов онлайн-общения как с родственниками и

друзьями, так и с деловыми партнерами.

В постоянном огромном потоке информации бывает довольно сложно для себя структурировать огромное количество интернет-источников и исходящей от них информации. Таковыми источниками являются новостные сайты, интернет-трансляции и, набирающие популярность, новостные telegram-каналы.

1. **Обзор предметной области**
   1. **Теоретические сведения об интернет-ботах**

Telegram – это бесплатный мессенджер для смартфонов и персональных компьютеров под управлением всех наиболее распространенных на сегодняшний день операционных систем, позволяющий обмениваться не только текстовыми сообщениями, но и различными медиафайлами (картинки, музыка, архивы, текстовые документы). Функционально Telegram похож на другие мессенджеры, его главные преимущества перед конкурентами – скорость, защищенность, хранение данных в облаке (удаленный сервер) и бесплатность.

Свой трафик Telegram надежно зашифровывает, все вычисления производятся на удаленном сервере, а клиентская часть только шифрует данные и отправляет их на сервер. Для устойчивой работы был создан уникальный протокол MTProto, что существенно повысило безопасность и защиту от несанкционированной утери информации. При одновременном обмене сообщениями и фотографиями участников чата, конкуренты значительно уступают Telegram по скорости доставки информации.

Боты – это специальные программы, выполняющие различные функции и упрощающие жизнь их пользователей. Написанные для платформы Telegram, они предназначены для выполнения самых разных функций: от получения новостей до поиска информации и даже торговли акциями. Главное задачей бота является автоматический ответ после введенной ему пользователем команды. При этом, работая непосредственно через интерфейс Telegram, программа имитирует действия живого юзера, за счет чего пользование таким ботом гораздо удобно и понятно. [1]

Сами же боты делятся на несколько направлений:

• Чат-боты представляют из себя простейший чат, имитирующий общение на заданную пользователем тематику.

• Боты-информаторы – отдельный вид ботов, главная цель которых — информирование пользователя о тех или иных событиях (новости, мероприятия, публикации и т.п.).

• Игровые боты – это боты, с которыми можно поиграть в различные игры. По большей части, это текстовые версии разных игр времен Combats.ru

• Боты-асистенты – это боты, разработанные различными онлайнсервисами как дополнение к основной веб-версии. Алгоритм работы бот-утилит состоит в следующем: сообщения, команды и запросы, отправленные пользователями, передаются на программное обеспечение, запущенное на серверах разработчиков.

Посреднический анонимный сервер Telegram обрабатывает шифрование и осуществляет обратную связь между утилитой и пользователем. Взаимодействие между пользователем и ботом выглядит следующим образом: пользователь отдает команду боту, бот передает команду на сервер, после этого программа на сервере обрабатывает полученный от бота запрос, затем сервер отдает ответ боту, в итоге бот выводит ответ на экран приложения пользователю. И этот цикл повторяется раз за разом, когда пользователь взаимодействует с любым телеграм-ботом. [2]

Взаимодействие с серверами происходит с помощью простого HTTPSинтерфейса, который представляет собой упрощенную версию API Telegram. По-другому этот интерфейс можно назвать программным каталогом или боталгоритмом. Новые bot-утилиты создаются с помощью специальной утилиты @BotFather.

* 1. **Назначение приложения**

Чаще всего, современный человек, который заходит в интернет не хочет увидеть какую-то определенную новость. Как правило, важен сам факт наличия таковой новости. Разработанный бот позволяет получить «быструю» новость в одной из 4 категорий: Текущая погода, Политика, Погода и Случайная новость, представлены на рисунке 1.

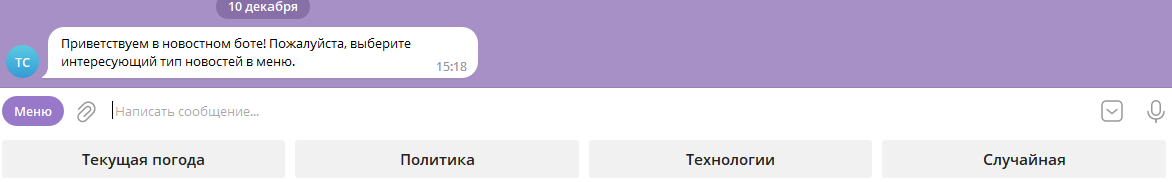


Рисунок 1 – Категории новостей

Разработанное приложение представляет собой программу для взаимодействия с Telegram API. Данное приложение позволяет получить «быструю» новость а также просмотреть погоду.

* 1. **Область применения и условия функционирования**

Разработанное приложение может применяться в любом мобильном устройстве, содержащем приложение Telegram, что значительно расширяет круг его пользователей.

Для корректной работы приложения необходимо иметь на своем мобильном устройстве операционную систему Android версии 8 и выше или IOS 10 и выше. Также необходимо наличие самого приложения Telegram для взаимодействия с API.

Далее пользователю необходимо перейти по пригласительной ссылке https://t.me/Typical\_Solutions\_bot, либо же самостоятельно найти бота в приложении по кодовому имени @Typical\_Solutions\_bot. Для этого достаточно ввести кодовое имя в поисковую строку приложения Telegram.

Со стороны сервера, принимающего запросы от Telegram API, должна быть представлена возможность для функционирования языка PHP версии 7.0 и выше. В разработанном приложении разработана функциональность как для работы в веб-хуком, так и с приложением, развернутым локально.

1. **Обзор аналогов**

Требования к сервису должны содержать наиболее эффективные для данной задачи решения. Из этого следует, что при изучении и сравнении сервиса с аналогами основными задачами можно выделить четкое формулирование требований к своему проекту и нахождение наиболее успешных решений в уже существующих разработках. Изучение уже разработанных конкретных аналогов, рассмотрение их достоинств и недостатков, маркетинговых моделей, технических решений, поможет лучше понять, каким должен быть создаваемый сервис, какие элементы необходимо включить, а в каких нет надобности, сформулировать требования к проекту.

Для этого был проведен анализ различных сервисов на основании их собственного позиционирования на собственных платформах, статей и отзывов о них простых пользователей, личного опыта использования. Основными критериями были готовая версия продукта, востребованность у целевой аудитории потребителей, различные технические или маркетинговые решения сервисов для изучения нескольких подходов к реализации продукта.

Под данные критерии подошли следующие проекты: ControllerBot, Combot, Channely. ControllerBot.

ControllerBot является самым популярным ботом для публикации контента. В его функционал входят: отложенная публикация, добавление комментариев под публикацией, отложенное удаление публикации. К его преимуществам можно отнести:

• Простой и понятный интерфейс

• Бессрочную бесплатную версию

Также имеется ряд серьезных недостатков:

• Небольшой функционал

• Отсутствие web-версии

• Не всегда работает стабильно

• Нет возможности опубликовать фотографии в формате “Альбом”

Далее рассмотрим еще один поппулярный бот: Combot.

Среди его возможностей — детальная аналитика, настройки модерации, система репутации, триггеры, предупреждения, а также система антиспам и различные отчеты.

Преимущества:

• Есть бесплатный тариф до 200 подписчиков

• Обилие функций и настроек

• Глубокая статистика по каналу

• Наличие мобильной версии

Недостатки:

• Минимальный платный тариф 9.99$ в месяц

• Сложный интерфейс

Из-за ограничений в интерфейсе ботов самого Телеграма, наличие web-интерфейса значительно облегчает взаимодействие пользователя с продуктом, давая возможность видеть и взаимодействовать с большим количеством параметров и опций, чем в мессенджере. Возможность настроить собственный интерфейс в web-версии позволяет сделать его более удобным и понятным для широкой аудитории пользователей.

Многие Telegram-боты имеют функционал, необходимый для управления медийными каналами, но не требующийся простым пользователям для организаций конференций, локальных сообществ. Наличие большого количества функций может осложнять интерфейс для восприятия обычных пользователей, поэтому необходимо сразу определить целевую аудиторию, на которую рассчитан проект.

1. **Модель прецедентов**

Взаимодействие с разработанной программой можно описать с помощью схемы, представленной на рисунке 2.

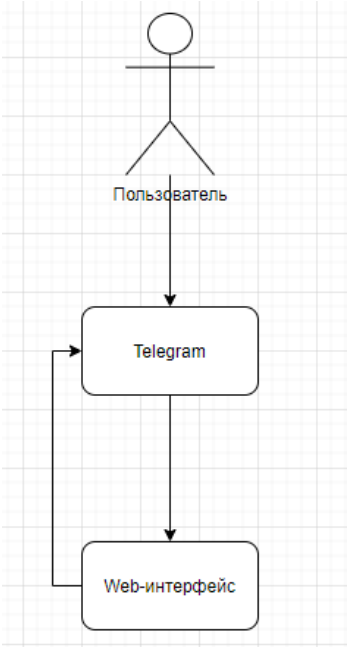


Рисунок 2 – Принцип взаимодействия с ботом

Пользователь с помощью установленного на своем мобильном устройстве приложения Telegram обращается к Web-интерфейсу, который, непосредственно, отправляет и получает запросы на сервер. Взаимодействие сервера и Web-интерфейса представлено на рисунке 3.

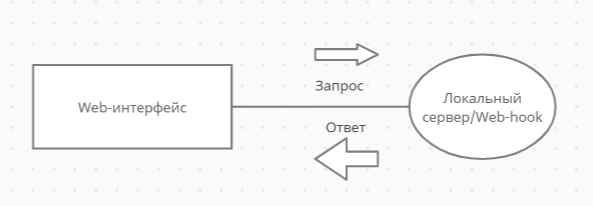


Рисунок 3 – Принцип отправки/получения запроса

Как видно на рисунке выше, запрос и ответ напрямую отправляются на сервер. В случае разработанного приложения Web-интерфейсом служит Telegram API – интерфейс, встроенный в приложение и имеющий методы, которые напрямую позволяют использовать весь доступный инструментарий для разработки бот-решений.

Процесс работы приложения выглядит следующим образом. При первом входе в бот, пользователю предлагается выбрать интересующую категорию новостей, представлено на рисунке 1. Далее, после выбора соответствующей категории, пользователь получает одну новость, представлено на рисунке 4.

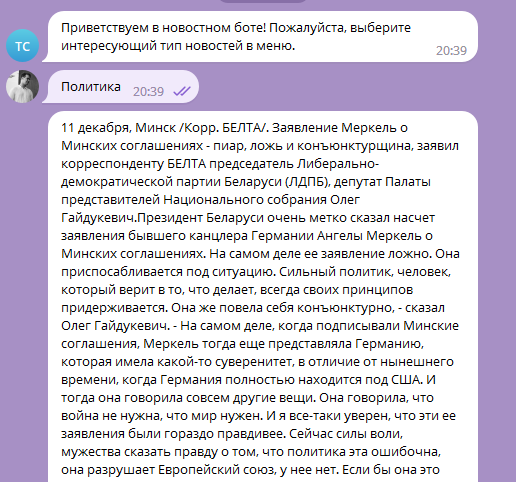


Рисунок 4 – Новость из категории «Политика»

При нажатии на три функциональные кнопки: «Политика», «Технологии», «Случайная новость» появится одна из новостей в соответствующей категории. Каждая новость соответствует новости с сайта <https://www.belta.by/>.

При нажатии на кнопку «Текущая погода», пользователь получит текущее значение температуры воздуха, влажности, скорости ветра и видимости. Вывод сообщения представлен на рисунке 5.

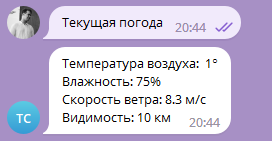


Рисунок 5 – Вывод текущей погоды

Каждое значение текущей погоды соответствует значению с сайта <https://world-weather.ru/pogoda/>.

1. **Функциональная модель приложения**

Архитектура данного приложения построена таким образом, чтобы соответствовать модели MVC – Model-View-Controller. Это схема разделения данных приложения и управляющей логики на три отдельных компонента: модель, представление и контроллер — таким образом, что модификация каждого компонента может осуществляться независимо [3]. Визуализация архитектуры MVC представлена на рисунке 6.

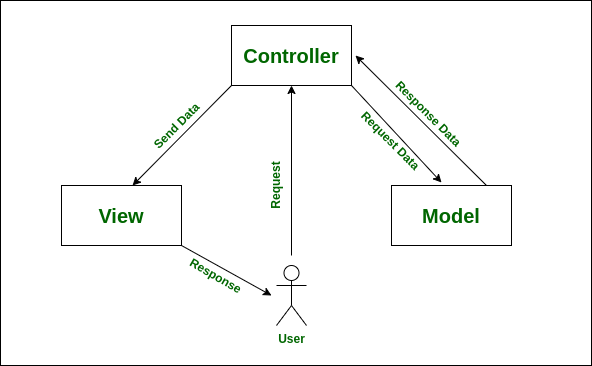


Рисунок 6 - Визуализация архитектуры MVC.

Данная архитектура была выбрана с расчетом на то, что приложение будет развиваться и дорабатываться. В случае разработанного приложения разработчик может воздействовать на компонент Model. Компоненты View и Controller содержит в себе Telegram API, так что фактическая схема приложения будет выглядеть немного иначе, представлена на рисунке 7.

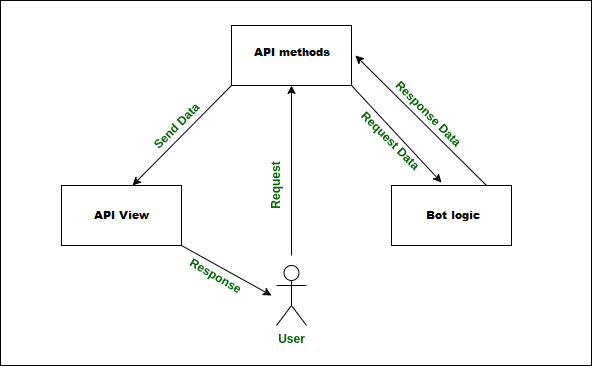


Рисунок 7 – Фактическая архитектура приложения

В соответствии с логикой архитектуры, директории, в которых расположены классы, имеют соответствующее название, представлены на рисунке 8.

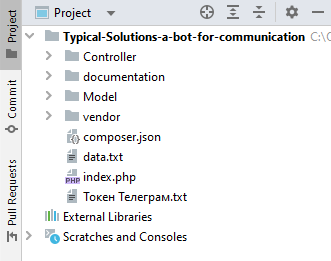


Рисунок 8 – Структура проекта

В директории «Controller» хранится единственный класс категории контроллер – FrontController, представлен на рисунке 9.

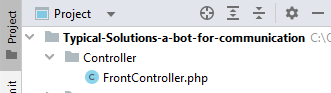
\

Рисунок 9 – класс типа контроллер

Этот класс отвечает за прием запросов и ответов, а также принимает решение, какую модель в дальнейшем будет использовать приложение для вывода пользователю соответствующей информации. Основной метод называется makeRoute(), представлен на рисунке 10.



Рисунок 10 – Метод makeRoute()

При вызове этого метода программа получает последнее отправленное пользователем сообщение и, в зависимости от типа этого запроса, вызывает конструктор соответствующего класса. Все модели хранятся в директории Model и представлены на рисунке 11.

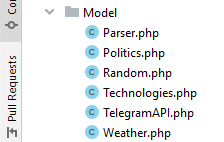


Рисунок 11 – классы типа Model.

Рассмотрим класс Technologies, представлен на рисунке 12. Конструктор этого класса вызывается при нажатии пользователя на соответствующую кнопку.



Рисунок 12 – Класс Technologies.

Данный класс работает следующим образом. Получает ссылку на сайт, откуда будет взята текстовая информация. Затем с помощью класса Parser и его методов, используя методы PHP и регулярные выражения, преобразует код в массив ссылок. Затем случайно выбирается случайная новость, которая снова обрабатывается методами класса Parser и сразу же отправляется пользователю с помощью метода Telegram API sendRequest(). Подобным образом работают и остальные модели.

Рассмотрим остальные папки проекта. В папке «vendor», представленной на рисунке 13, хранятся необходимые зависимости, которые устанавливаются с помощью утилиты Composer.

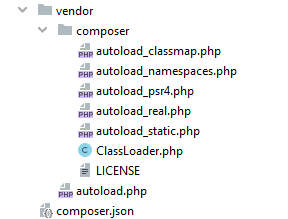


Рисунок 13 – Директория vendor

Данная организация подключения зависимостей позволяет подключать и использовать методы и конструкторы разных классов внутри друг друга без явного объявления, что в дальнейшем устранит некоторые трудности при расширении функционала данного решения. Достаточно добавить необходимые директории или классы в файл composer.json, представленном на рисунке 14.



Рисунок 14 – Конфигурационный файл composer.json

1. **Заключение**

В настоящем курсовом проекте изучен и рассмотрен процесс разработки приложения для взаимодействия с Telegram API на примере Telegram-бота. Также было разработан и развернут полноценный новостной бот с соответствующим функционалом. В данном проекте была учтена возможность расширения и доработки и для этого была использована архитектура MVC и управление зависимостями с помощью Composer, что в дальнейшем будет способствовать упрощению процесса улучшения функциональности данного приложения.

**Литература**

1. Wikipedia [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki Бот\_(программа). Дата доступа: 14.10.2022
2. Toledo L. python-telegram-bot Documentation. [Электронный ресурс]. Режим доступа: https://readthedocs.org/projects/pythontelegram-bot/downloads/pdf/latest/. Дата доступа: 18.11.2022.
3. MVC для веб [Электронный ресурс]. Режим доступа: <https://habr.com/ru/post/181772/>. Дата доступа: 10.10.2022

**Приложение 1**

**Листинг кода:**

**<?php***/\*\*  
 \** ***@author*** *Anton Naumov  
 \*/***class** FrontController{  
  
 **public static function** getInstance()  
 {  
 **static** $instance;  
 **if** (!**isset**($instance)) $instance = **new self**;  
 **return** $instance;  
 }  
  
 **public function** makeRoute(){  
  
 $TelegramAPI = **new** TelegramAPI();  
  
 *sleep*(10);  
 $chatUpdates = $TelegramAPI->getUpdates();  
 $lastMessage = $chatUpdates[**"result"**][*count*($chatUpdates[**"result"**])-1][**"message"**][**"text"**]; *//последнее текстовое сообщение* **switch** ($lastMessage){  
 **case 'Политика'**: **new** Politics();  
 **break**;  
 **case 'Текущая погода'**: **new** Weather();  
 **break**;  
 **case 'Случайная'**: **new** Random();  
 **break**;  
 **case 'Технологии'**: **new** Technologies();  
 **break**;  
 **default**: $TelegramAPI->sendRequest(**'sendMessage'**, [**'chat\_id'** => ***CHAT\_ID***, **'text'** => **'Ваше сообщение не распознано. Убедитесь, что точно попали по кнопке'**]);  
 }  
 }  
}

**<?php  
  
class** Parser{  
  
 **public static function** parse($p1, $p2, $p3) {  
 $num1 = *strpos*($p1, $p2);  
 **if** ($num1 === **false**) **return** 0;  
 $num2 = *substr*($p1, $num1);  
 **return** (*substr*($num2, 0, *strpos*($num2, $p3)));  
 }  
  
 **public static function** parseFirstLink($pageWithLinks){  
 *preg\_match\_all*(**'#<a [^>]\*href="(.\*)"[^>]\*>#Ui'**, $pageWithLinks, $matches);  
 **return** $matches[1][0];  
 }  
}

**<?php  
  
class** Politics  
{  
 **public function** \_\_construct(){  
  
 $content = *file\_get\_contents*(**'https://www.belta.by/politics/'**);  
  
 $pageWithLinks = Parser::*parse*($content, **'<div class="main\_in\_rubric">'**, **'</div>'**);  
 $pageLink = Parser::*parseFirstLink*($pageWithLinks);  
 $page = *file\_get\_contents*($pageLink);  
  
 *preg\_match\_all*(**'#<p>(.+?)</p>#is'**, $page, $arr);  
 $text = *implode*($arr[0]);  
  
 $text = *str\_replace*([**'<p>'**, **'</p>'**], **''**, $text);  
 $text = *preg\_replace*(**"/[a-z]/i"**, **""**, $text);  
 $text = *strtr*($text, [**'<'**=>**''**, **'>'**=>**''**, **':'**=>**''**, **'='**=>**''**, **'\_'**=>**''**, **'&'**=>**''**, **'"'**=>**''**, **'"читать далее"'**=>**''**]);  
 $text = *trim*($text);  
  
 TelegramAPI::*sendRequest*(**'sendMessage'**, [**'chat\_id'**=>***CHAT\_ID***, **'text'**=>$text]);  
  
 }  
  
}

**<?php  
  
class** Random{  
 **public function** \_\_construct(){  
 TelegramAPI::*sendRequest*(**'sendMessage'**, [**'chat\_id'**=>***CHAT\_ID***, **'text'**=>**'Случайная новость в том, что она случайная'**]);  
 }  
}

**<?php  
  
class** Technologies{  
 **public function** \_\_construct(){  
  
 $content = *file\_get\_contents*(**'https://www.belta.by/tech/ '**);  
  
 $pageWithLinks = Parser::*parse*($content, **'<div class="main\_in\_rubric">'**, **'</div>'**);  
 $pageLink = Parser::*parseFirstLink*($pageWithLinks);  
 $page = *file\_get\_contents*($pageLink);  
  
 *preg\_match\_all*(**'#<p>(.+?)</p>#is'**, $page, $arr);  
 $text = *implode*($arr[0]);  
  
 $text = *str\_replace*([**'<p>'**, **'</p>'**], **''**, $text);  
 $text = *preg\_replace*(**"/[a-z]/i"**, **""**, $text);  
 $text = *strtr*($text, [**'<'**=>**''**, **'>'**=>**''**, **':'**=>**''**, **'='**=>**''**, **'\_'**=>**''**, **'&'**=>**''**, **'"'**=>**''**, **'-'**=>**' '**, **'/'**=>**' '**, **'?'**=>**''**]);  
 $text = *trim*($text);  
  
 TelegramAPI::*sendRequest*(**'sendMessage'**, [**'chat\_id'**=>***CHAT\_ID***, **'text'**=>$text]);  
 }  
}

**<?php***/\*\*  
 \** ***@author*** *Anton Naumov  
 \*/***const *TOKEN*** = **'5651939402:AAEkpC2SVaxeGanut29D\_VlGbNM\_nYKNfqQ'**;  
**const *BASE\_URL*** = **'https://api.telegram.org/bot'**;  
**const *CHAT\_ID*** = 445503956;  
  
**class** TelegramAPI  
{  
 **public function** \_\_construct()  
 {  
 $this->sendRequest(**'sendMessage'**, [**'chat\_id'** => ***CHAT\_ID***, **'text'** => **'Приветствуем в новостном боте! Пожалуйста, выберите интересующий тип новостей в меню.'**,  
 **'reply\_markup'** => *json\_encode*(**array**(  
 **'keyboard'** => **array**(  
 **array**(  
 **array**(  
 **'text'** => **'Текущая погода'**,  
 **'url'** => **''**,  
 ),  
 **array**(  
 **'text'** => **'Политика'**,  
 **'url'** => **''**,  
 ),  
 **array**(  
 **'text'** => **'Технологии'**,  
 **'url'** => **''**,  
 ),  
 **array**(  
 **'text'** => **'Случайная'**,  
 **'url'** => **''**,  
 ),  
 )  
 ),  
 **'one\_time\_keyboard'** => **TRUE**,  
 **'resize\_keyboard'** => **TRUE**,  
 )),]);  
 }  
  
 **public static function** sendRequest($method, $data, $headers = [])  
 {  
 $curl = *curl\_init*();  
 *curl\_setopt\_array*($curl, [  
 ***CURLOPT\_POST*** => 1,  
 ***CURLOPT\_HEADER*** => 0,  
 ***CURLOPT\_RETURNTRANSFER*** => 1,  
 ***CURLOPT\_URL*** => ***BASE\_URL*** . ***TOKEN*** . **'/'** . $method,  
 ***CURLOPT\_POSTFIELDS*** => *json\_encode*($data),  
 ***CURLOPT\_HTTPHEADER*** => *array\_merge*(**array**(**"Content-Type: application/json"**))  
 ]);  
 $result = *curl\_exec*($curl);  
 *curl\_close*($curl);  
 **return** (*json\_decode*($result, 1) ? *json\_decode*($result, 1) : $result);  
 }  
  
 **public function** getUpdates()  
 {  
 **return** $this->sendRequest(**'getUpdates'**, []);  
 }  
}

**<?php  
  
class** Weather{  
 **public function** \_\_construct(){  
  
 $content = *file\_get\_contents*(**'https://world-weather.ru/pogoda/belarus/minsk/'**);  
  
 $temperature = Parser::*parse*($content, **'<div id="weather-now-number">'**, **'</div>'**); *//+* $temperature = *preg\_replace*(**"/[a-z]/i"**, **""**, $temperature);  
 $temperature = *strtr*($temperature, [**'<'**=>**''**, **'>'**=>**''**, **':'**=>**''**, **'='**=>**''**, **'\_'**=>**''**, **'&'**=>**''**, **'"'**=>**''**, **'--'**=>**''**, **'-'**=>**''**, **'/'**=>**''**]);  
  
 $text = **'Температура воздуха: '** . $temperature . ***PHP\_EOL*** . **'Влажность: '** . $humidity . ***PHP\_EOL*** . **'Скорость ветра: '** . $wind . ***PHP\_EOL*** . **'Видимость: '** . $visibility;  
  
 TelegramAPI::*sendRequest*(**'sendMessage'**, [**'chat\_id'**=>***CHAT\_ID***, **'text'**=>**"**$text**"**]);  
 }  
}

**<?php***/\*\*  
 \** ***@author*** *Anton Naumov  
 \*/***require\_once "vendor/autoload.php"**;  
FrontController::*getInstance*()->makeRoute();

{  
 **"name"**: **"app/app"**,  
 **"description"**: **"app"**,  
 **"minimum-stability"**: **"stable"**,  
 **"require"**: {  
 **"ext-mysqli"**: **"\*"** },  
 **"autoload"**: {  
 **"psr-4"**: {  
 **""**: [  
 **"Controller/"**,  
 **"Model"**,  
 **"Controller/FrontController.php"** ]  
 }  
 }  
}