Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
Информационные системы

УТВЕРЖДАЮ
Зав. кафедрой ИС
_____ П.П. Дьячук
подпись инициалы, фамилия
« ___» _____ 2019г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 Информационные системы и технологии

Разработка бота-редактора для групп ВКонтакте

Руководитель	_____ подпись, дата	доцент, к.т.н.	И.А. Легалов
Выпускник	_____ подпись, дата		И.В. Галуза
Нормоконтролер	_____ подпись, дата	ст. преподаватель	Ю. В. Шмагрис

Красноярск 2019

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
Информационные системы

УТВЕРЖДАЮ
Зав. кафедрой ИС
_____ П.П. Дьячук
подпись инициалы, фамилия
« ___» _____ 2019г.

ЗАДАНИЕ

НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

в форме бакалаврской работы

Студенту Галуза Ивану Васильевичу

Группа КИ15-13Б Направление 09.03.02

Информационные системы и технологии

Тема выпускной квалификационной работы: «Разработка бота-редактора для групп ВКонтакте».

Утверждена приказом по университету № 7237/с от 24.05.2019

Руководитель ВКР: И.А. Легалов, к.т.н., доцент кафедры «Информационные системы» ИКИТ СФУ.

Исходные данные для ВКР: Требования к разрабатываемому приложению, рекомендации руководителя.

Перечень разделов ВКР: Общие сведения, введение, описание предметной области, алгоритмы популярности, технологии разработки, требования к программе, среда разработки, система управления базами данных, библиотека Synapse и REST API, ВКонтакте API, программная реализация, заключение, список использованных источников.

Перечень графических материалов: Презентация, выполненная в Microsoft Office PowerPoint 2016

Руководитель ВКР _____ И.А. Легалов
подпись

Задание принял к исполнению _____ И.В. Галуза
подпись

« ___» _____ 2019г.

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка бота-редактора для групп ВКонтакте» содержит 56 страниц текстового документа, 28 рисунков и 26 использованных источника.

ПРИЛОЖЕНИЕ, БОТ, АВТОМАТИЗАЦИЯ, КОНТЕНТ-МЕНЕДЖЕР, ВКОНТАКТЕ.

Целью выпускной квалификационной работы является автоматизация работы контент-менеджера для групп ВКонтакте.

Для достижения поставленной цели были поставлены следующие задачи:

- Изучить предметную область;
- Разработать алгоритм вычисления интересных для пользователей тем;
- Разработать программу для автоматического поиска и публикации контента на основе выявленных интересов текущих участников.

В результате выпускной квалификационной работы было разработано приложение, автоматизирующее и ускоряющее процесс подбора и публикации контента в группе ВКонтакте.

СОДЕРЖАНИЕ

1 Общие сведения

1.1 Введение

С появления первого подобия интернета прошло более 50-и лет. Однако уже сейчас трудно представить нашу жизнь без него. Интернет не только быстро ворвался в нашу жизнь, но и глубоко в ней осел. Постепенно люди, иногда сами того не замечая, перенесли свою жизнь в виртуальное пространство. В том числе и взаимодействие друг с другом. Большая часть этих взаимодействий происходит в местах наибольшего скопления виртуальных личностей – в социальных сетях, например, ВКонтакте, Facebook, Twitter или Instagram.

Социальные сети преподносят своим пользователям огромные возможности для взаимодействия: мгновенные сообщения, публикация информации о себе и различных событиях, видео и фотографии, объединение в группы по интересам, информация о различных товарах и услугах, а также о мероприятиях. Они имеют огромное влияние на абсолютно любые сферы жизнедеятельности. Благодаря им популяризируются товары, компании, люди и многое другое, однако из-за них же эта же популярность может быть и потеряна.

Влиянием социальных сетей можно воспользоваться для продвижения собственных товаров и услуг, однако для этого необходимо заинтересовать других пользователей. Это и есть самая большая проблема, которую необходимо решить.

Так разрабатываемый интернет-робот, именуемый «VKGroupRedactor» предназначен для решения проблем поиска и публикации контента с наиболее интересной для аудитории тематикой и автоматизацией этого процесса.

Цель работы - автоматизация работы контент-менеджера для групп ВКонтакте.

Для достижения поставленной цели необходимо решить ряд следующих задач:

• Изучить предметную область;

• Разработать алгоритм вычисления интересных для пользователей тем;

• Разработать программу для автоматического поиска и публикации контента на основе выявленных интересов текущих участников;

1.2 Описание предметной области

БОТ-редактор VKGroupRedactor должен уметь работать с социальной сетью ВКонтакте, с её сообществами, постами и дополнительной информацией, предоставляемой выбранной площадкой. Так перед разработкой необходимо разобраться со всеми уровнями работы и понятиями, повсеместно использующимися пользователями как ВКонтакте, так и других соц. сетей.

1.2.1 Основные понятия

API – интерфейс программирования приложений, использующийся для упрощения работы со сторонними сервисами посредством предоставления готового функционала.

Авторизация – процесс предоставления одному или нескольким лицам прав на выполнение определенных действий.

Активность – действие пользователя: лайк, репост или комментарий.

Бот – специальная программа, выполняющая в автоматическом режиме какие-либо действия и имитирующая деятельность человека.

Контент – любое значимое наполнение информационного ресурса, включающее в себя как текст, так и мультимедийную информацию (изображения, видео, прикрепленные документы).

Контент-менеджер – человек, занимающийся размещением контента на информационном ресурсе.

Лайк – способ и функция взаимодействия пользователя с контентом ресурса путем выражения своего одобрительного мнения. С англ. Like - нравится.

Пост – короткое сообщение, оставленное на стене личного аккаунта или группы в социальной сети.

Постинг – процесс размещения постов в сообществе.

Пользователь – человек или программа, использующая функционал ресурса.

Репост – действие, направленное на распространение понравившегося поста.

Сообщество ВКонтакте – виртуальное объединение пользователей по каким-либо интересам в рамках социальной сети.

Социальная сеть – Интернет-ресурс, предназначенный для создания и отражения социальных взаимоотношений в виртуальном пространстве.

Токен – специальный ключ доступа, хранящий в себе информацию о правах доступа профиля в зашифрованном виде.

1.2.2 Сообщества, их виды и роли

В социальной сети ВКонтакте огромное количество аккаунтов людей с различными характерами, мировоззрениями, целями и интересами. Ни для кого не секрет, что людям со схожими качествами всегда было интересно держаться рядом, поэтому они объединяются в виртуальные группы в рамках этой платформы, называющиеся «сообществами», где они делятся мыслями, общаются или обсуждают различные события.

На первый взгляд все это выглядит несерьезно, однако стоит понимать, что такие группы насчитываю сотни, тысячи, а иногда и миллионы пользователей, тогда такие сообщества становятся мощным инструментом маркетинга. Ведь немало компаний хотят заявить о себе, о своих продуктах и услугах на

эту огромную аудиторию. Таким образом эти объединения становятся отличной возможностью найти новых клиентов для многих коммерческих организаций.

ВКонтакте существует несколько видов сообществ, реализующих различные цели: обмен мнениями, распространение новостей и информации или организация концертов и вечеринок. Все они имеют свои достоинства и недостатки.

Группа – это самый универсальный формат сообществ, обеспечивающий множество функций. Этот вид сообществ используется в первую очередь для объединения людей по интересам, но также для рекламирования и продаж услуг и продуктов различных компаний.

Особенности группы:

- пользователи могут помогать или сами создавать контент: добавлять видео- и фотоматериалы, делать заметки и сообщения к постам;
- несколько уровней закрытости: открытые, закрытые и частные. К группам первого типа получают доступ любые пользователи. Второго: те, чью заявку одобрила администрация. Третьего: только по приглашению администратора;
- большое количество тематической информации, в которой легко сориентируется любой участник;
- возможность удаления участников.

Таким образом группа – это отличный вариант распространения тематического контента, новостей, акций, сопровождающихся фотографиями, опросами и видеоматериалами.

Публичная страница

Контент таких сообществ могут просматривать все пользователи мировой паутины, даже незарегистрированные во ВКонтакте. Таким образом публичные страницы используются в основном для продвижения брендов как на

рисунке 1, новостей, услуг и многое другое. В них часто появляются рекламные записи и, как правило, существуют партнерские программы, являющиеся одним из основных доходов.

Особенности публичной страницы:

- отображается в списках страниц всех участников при публикации интересного контента, что увеличивает охват аудитории;
- публиковать контент может только администрация страницы, однако участники могут предлагать свои материалы;
- информация сообщества, как и указанные вами контактные данные, находятся в «шапке» страницы;
- простота раскрутки, так как люди чаще подписываются на сообщества такого типа.

Рисунок 1 – Публичная страница спортивного бренда Nike

Мероприятия – вид сообщества, созданный для информирования участников о различных событиях и мероприятиях. В целом, это та же группа, имеющая небольшие изменения. Пример такой группы представлен на рисунке 2 нижу. Руководитель сообщества – организатор, которым может являться аккаунт человека или целое сообщество. У мероприятия обязательно установлено время начала и окончания события, которое можно изменить при необходимости. У нового участника есть выбор ответа: «Точно пойду», «Возможно пойду» и «Не могу пойти».

Особенности мероприятия:

- участникам всегда приходит напоминание о приближении события;
- в сообщество можно приглашать до 5000 участников ежедневно;

• имеется возможность смены даты и времени при переносе мероприятия.

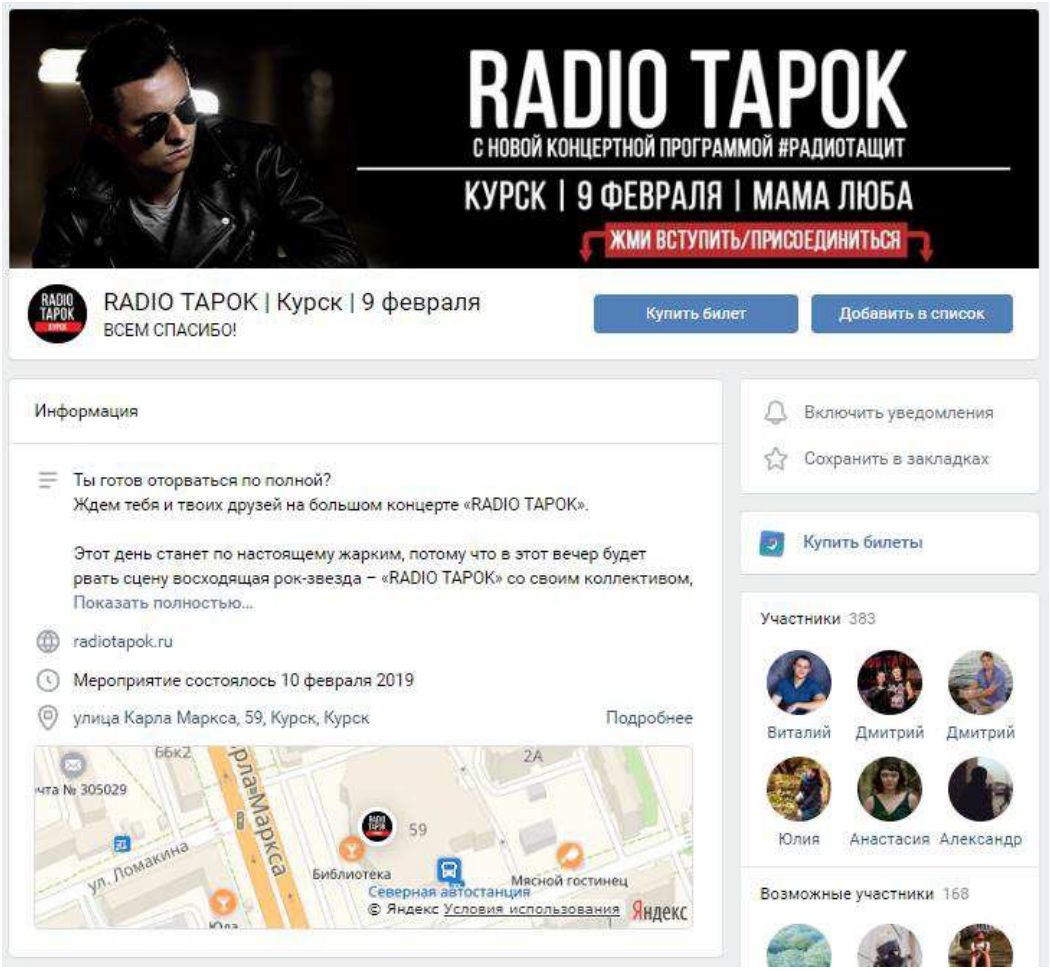

Рисунок 2 – Страница мероприятия с сопутствующей информацией

1.2.3 Роль контент-менеджера в сообществах

Контент-менеджер (редактор) – специалист, занимающийся созданием контента для сайтов, групп и других информационных ресурсов.

Эта профессия появилась почти сразу с появлением интернета и со временем становится все популярнее так как количество сайтов и информационных ресурсов с каждым днем все больше и больше. Так еще в 2010 году количество сайтов не превышало и 255 миллионов, в 2012 их стало в три раза больше - 634 миллиона, а в 2015 – более 1-ого миллиарда.

Работа контент-менеджера варьируется и зависит от площадки, на которой он работает, например, при работе над сайтом этому специалисту требуется следить за структурой сайта, публикацией мультимедийной информацией, контролировать SEO-продвижение и многое другое, однако в такой социальной сети, как ВКонтакте, задач не так уж много: подготовка и публикация постов, общение с участниками сообщества, популяризация групп и, иногда, контроль комментариев и спама, а так же мультимедийного материала, публикуемого подписчиками.

Таким образом, чем больше задач решает редактор, тем больше его заработная плата. В среднем по запросу «контент-менеджер для групп ВКонтакте» на сайте Premium-job.ru можно получить такие результаты, представленные на рисунке 3.

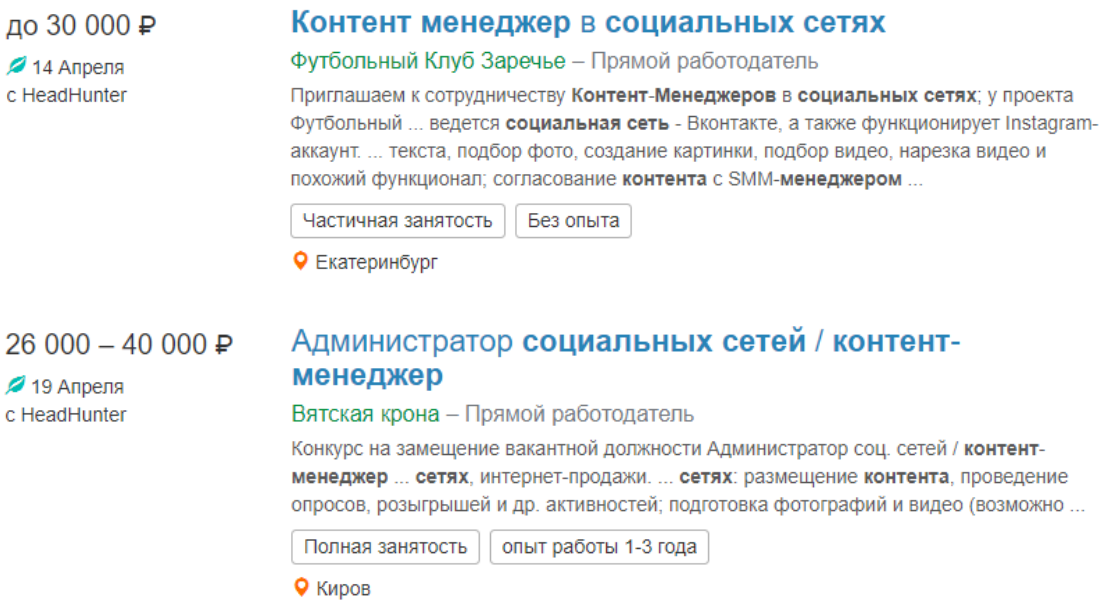

Рисунок 3 – Вакансии контент-менеджеров социальных сетей

В обязанности контент-менеджера страниц в социальных сетях обычно входят:

- разработка контент-плана с учетом стратегии продвижения компании;
- подготовка и публикация материала в заранее оговоренное время;

- ответы на вопросы и комментарии участников;
- создание, настройка и размещение рекламы.

Однако стоит помнить, что каждый работодатель вправе сократить или дополнить список этих обязанностей.

1.2.4 Боты и их значение

На просторах интернета огромное количество процессов выполняют боты – программа-робот, созданная для выполнения однотипных действий, которые вручную человеку делать долго или тяжело из-за однообразности повторяемых действий. Такая программа способна выполнять бесконечное количество раз одно и то же действий и никогда не уставать.

Так в социальных сетях ботов используют для продвижения аккаунтов и групп, имитируя наличие огромной аудитории: комментарии, лайки, новые участники и другая активность. Использовать их очень выгодно для предпринимателей и владельцев сообществ практически любой направленности, ведь с популярных групп можно иметь отличный пассивный доход.

Исходя из вышеперечисленного можно сделать вывод, что роль интернет-робота достаточно велика в 21-м веке, особенно важна его продуктивность: порой он один может сделать больше, чем сто человек. Также боты принимают участие в самых разных сферах жизнедеятельности и выполняют развлекательные, коммерческие, информационные и другие функции. В связи с этим существует несколько их видов:

- Боты-продавцы;
- SEO-боты;
- CPA боты-многодневки;
- Политические и социальные боты.

Боты-продавцы

Сейчас все больше людей приобретают тот или иной товар в интернет-магазинах, посещая при этом не одну тройку информационных ресурсов. При этом пользователь может столкнуться с ситуацией, когда, после просмотра одного из сайтов, во ВКонтакте ему придёт сообщение от аккаунта того самого сайта с сообщением: «Добрый день, я заметил, что Вы посещали наш сайт, но ничего не заказали. Вам что-нибудь подсказать?». Это и есть бот-продавец, действующий по достаточно распространенной схеме, для которой некоторые компании используют специальные сервисы по отслеживанию профилей из социальных сетей. Цель таких программ проста и понятна – продажа продукции или услуг компании. Отличительная черта таких программ – схожесть с сервисом-владельцем.

SEO-боты – боты, занимающиеся накруткой основных показателей аккаунтов, групп в социальных сетях и даже на площадках голосований. Под основными показателями подразумеваются параметры аккаунтов, групп и товаров, выдаваемые системой, в конкретном случае социальной сетью ВКонтакте, в самом начале при регистрации. Это могут быть: количество друзей у аккаунта, подписчиков и участников сообществ, лайков, комментариев и репостов.

Таких ботов остаточно просто отличить от профилей реальных людей. Это, как правило, пустые аккаунты, состоящие в большом количестве групп и имеющие на своей странице множество репостов, при почти полном отсутствии своих записей.

CPA боты – это программы, похожие на продавцов, однако действуют по гораздо боле сложным алгоритмам. Стоит сказать о том, что владельцы таких ботов заботятся о них на начальном этапе: заполняют профиль записями, фотографиями, добавляют друзей и подписчиков, если это необходимо, совершают множество других действий по развитию. Важной особенностью таких интернет-роботов является не только наполненность аккаунта, но и их разделение на необходимые классы продвигаемых продуктов.

Например, боты, чьи аккаунты наполнены огромным количеством привлекательных моделей автомобилей марки BMW, подписаны на соответствующие сообщества и содержат рекламу этого бренда, явно предназначены для его раскрутки и продвижения. Такие боты будут не только состоять в тематических группах, но и вести активную деятельность по привлечению внимания к себе и, соответственно, к бренду посредством репостинга, комментирования, явной и скрытой рекламы.

Такие боты могут легко по той же схеме продвигать все что угодно, например, «боулинг-клуб в Красноярске». Для этого его достаточно подписать на соответствующие группы и все. Таким программам крайне легко сменить вид деятельности или его направление.

Отличительной чертой такого типа ботов может быть наличие ссылки на другой информационный ресурс, не относящийся к текущей платформе, с какой-либо характерной рекламной подписью для перехода, например, «еще больше информации тут (ссылка)».

Политические социальные боты – пожалуй, самая сложная категория автоматизированных программ. Их могут растить месяцами и даже годами и использовать, например, в политических интересах. Они почти на сто процентов имитируют реального пользователя, но несут скрытный характер своей деятельности, деятельности «лидеров мнений».

Например, эти боты могут максимально имитировать роль обычного учителя, который делится свои мнением и нередко публикует информацию, отличную от «основной» темы. Такие программы часто имеют широкий круг друзей, их контент разносится SEO ботами и может нести как скрытный, так и очевидный посыл.

Политические в разной степени распространены в большинстве стран, имеют свои цели и названия. Вот некоторые из них:

- «Тролли из Ольхино» - боты РФ, использующиеся для иллюзии высокой поддержки власти;

- «Порохоботы» - Украинские боты, созданные для поддержки президента;
- в США по рассказам журнала «Motherboard» социальных ботов использовали обе конкурирующие фракции кандидатов на президентский пост Дональда Трампа и Хиллари Клинтон.

1.3 Алгоритмы популярности

В мире нет ничего абсолютно одинакового - у всего есть как свои плюсы, так и минусы. В совокупности эти показатели способствуют формированию общего мнения о предмете, человеке или событии. Такое мнение очень важно для нас, так как благодаря ему мы можем сравнивать один объект с другим похожим и выбирать наиболее привлекательный. Так, выбирая новый пылесос в магазине, покупатель обязательно обратит внимание на такие показатели, как мощность всасывания, объем бака, наличие щетки и многое другое. Наличие или разница этих показателей сформирует у него некоторый рейтинг для каждого пылесоса по отношению к другим, благодаря которому покупатель выберет тот, что ему больше подходит.

Интересно, что в интернете люди так же формируют рейтинг или меру популярности для товаров, услуг, компаний, продавцов и это далеко не полный список. Причем в интернет-магазинах популярность товара играет огромную роль, так как её формируют сами покупатели и чем она больше, тем больше доверие к нему и тем с большей вероятностью этот продукт будут покупать в будущем. По аналогии с популярностью товаров в магазинах работает популярность статей на новостных сайтах, фотографов на тематических форумах, компаний, представляющих свои интересы в интернете.

С одной стороны, такой важный показатель, как оценка, достаточно прост в своей реализации. Ведь что может быть проще, чем выставить условный «-» или «+» к товарам и отсортировать их. Однако не все так просто.

Одной из основных задач разрабатываемого бота является разработка алгоритма для расчета рейтинга постов, позволяющего определять интерес участников к различным темам. Он должен решать, посты какой тематики публиковать в ближайшем будущем. Такая задача довольно популярна, и на сегодняшний день существует несколько рейтинговых алгоритмов, с которыми необходимо разобраться:

- Богатые богатеют – это алгоритм, показывающий темы, имеющие наиболее высокую оценку. Таким образом их популярность будет с каждым разом все выше и выше, не давай пробиться «молодняку»;
- Плюс/минус – это достаточно распространенный метод ранжирования, позволяющий как повышать оценку известности тем, так и понижать его;
- «Пять звезд» – очень популярная реализация оценки, выраженная, как правило, звездами с разной степенью наполненности, как на рисунке 4. Однако, на сколько бы ни был популярен такой метод, он имеет очень весомый недостаток – голос одного пользователя. Ведь рейтинг одного человека в «5 звезд» будет всегда выше 99-и человек в «4 звезды» и 1-ого в «5 звезд». Такого можно избежать, если игнорировать темы с малой активность. Но есть и другие решения данной проблемы – математическое ожидание рейтинга. Суть такого метода заключается в присвоении новой теме средней оценки всех тем.

Рисунок 4 – Метод ранжирования типа «Пять звезд»

2 Технологии разработки

Перед началом разработки ПО бот-редактор следует провести анализ требований, которые должны быть реализованы в приложении.

2.1 Требования к программе

По описанию поставленной задачи выделены следующие требования к разрабатываемому Боту-редактору:

2.1.1 Функциональные требования

- Обеспечить работу приложения с сервисом ВКонтакте по средствам ВКонтакте API;
- Обеспечить работу с несколькими профилями;
- Реализовать возможность выполнения нескольких задач для одного или нескольких профилей одновременно;
- Внедрить базовые функции добавления, сохранения и загрузки информации о профилях, задачах и логах программы;
- Реализовать функционал ведения журнала действий в программе.

2.1.2 Нефункциональные требования

- Операционная система для использования: Windows 8, 8.1, 10;
- Время отклика Бота-редактора не должно превышать пяти секунд;
- Интерфейс пользователя необходимо обеспечить диалоговыми окнами ошибок, оповещений, регистрации, добавления и редактирования информации;
- Обеспечить бесперебойное соединение с интернетом.

Перечень возможных аварийных ситуаций:

- Перебои электричества;
- Сбои Интернет-соединения;
- Ошибки пользователя;
- Ошибки сервиса ВКонтакте;
- Ошибки работы алгоритмов.

Способы предотвращения перечисленных ситуаций:

При каждом запуске программы необходимо проверять наличие Интернет-соединения. Для предотвращения возникновения бесконечных циклов на последнем этапе проектирования проводят тестирование на функциональность системы. Для решения ошибки потери данных при перебоях электричества, природных катаклизмов и пожаров возможна модификация системы в сетевой режим работы. База данных дублируется с периодичностью в 1 день, что позволит восстановить её при утрате данных.

По установленным требованиям были выбраны следующие технологии разработки:

- Среда разработки - Embarcadero RAD Studio Delphi XE10 с использованием кроссплатформенного фреймворка FireMonkey;
- База данных SQL Lite;
- Работа с сервисом ВКонтакте - VK API и библиотека Synapse для работы с безопасным протоколом HTTPS.

2.2 Среда разработки

В качестве среды разработки ПО была выбрана среда Delphi XE10 Berlin под выпуском Embarcadero RAD Studio.

Embarcadero RAD Studio – комплексное решение, позволяющее в относительно короткие сроки с минимумом усилий разрабатывать программные продукты благодаря концепции RAD.

2.2.1 Концепция RAD

RAD – концепция быстрой разработки программ, основанная на идее быстрого и удобного программирования.

Методология RAD со своей итеративной моделью развития проекта развивалась в противовес медлительному процессу разработки программного обеспечения 1970-ых годов, когда во время разработки успевали смениться требования к программе, что еще больше замедляло его. В 1980-ых годах Джеймс Мартин, сотрудник IBM и основатель этой концепции, сформулировал ее основные принципы:

- Основная цель – свести время разработки к минимуму;
- Ускоренная разработка прототипов, на которых следует уточнять требования заказчика;
- Цикличность разработки: каждая следующая версия продукта должна быть основана на предыдущей с исправлениями под требования заказчика;
- Минимизация временных затрат при разработке за счет дополнения функционала к уже готовым модулям предыдущей версии.

Исходя из принципа цикличности разработки ПО, его жизненный цикл имеет четыре этапа:

1. *Анализ* – разработчики совместно с пользователями обсуждают и определяют функциональные и нефункциональные требования, которые должен выполнять готовый продукт, расставляют для них приоритеты. Результатом этой фазы становится готовый список функциональных и нефункциональных требований к будущему продукту.

2. *Проектирование* – разработчики уточняют и, при необходимости, дополняю не выявленные требования, подробно рассматривают каждый процесс, разбивая его на подпроцессы, определяют требования к доступности данных. Результатом этапа являются:

   a. Информационная модель системы;
   b. Функциональные модели системы и её подсистем;

с. Первые версии экранов, отчетов, диалогов и других элементов взаимодействия с пользователем.

3. *Конструирование* – разработчики разных подгрупп итеративно создают первые прототипы модулей будущего приложения на основе моделей, полученных на предыдущем этапе. Заказчик и конечные пользователи занимаются оценкой результатов и вносят необходимые корректировки в функционал. В процессе разработки проходит и тестирование продукта. После окончания работы всех групп разработчиков происходит постепенная интеграция всех модулей в единый программный код с последующим тестированием этой совместной работы.

4. *Внедрение* – на этой фазе происходит внедрение новой системы одновременно работая с существующей, параллельно с обучением пользователей и организационными изменениями. Планирование этой фазы начинается уже на стадии проектирования.

В настоящее время эта концепция стала общепринятой при разработке ПО, в которых важны:

- Высокая скорость разработки;
- Низкая стоимость;
- Высокое качество.

Возможность использование RAD в проектах с такими характеристиками обеспечивается используемой в разработке итеративной модели жизненного цикла ПО, в котором продукт выпускается небольшими итерациями, наращивая свой функционал с каждым последующим этапом. При этом начальные версии не должны быть идеальными, главное – они должны быть рабочими. Так с каждой N-ой итерацией будущий продукт становится результативнее и ближе к намеченной цели.

Рисунок 5 – Наглядная интерпретация итеративной модели жизненного цикла

Наглядная интерпретация итеративной модели ЖЦ представлен на рисунке 5 с картиной Леонардо да Винчи «Мона Лиза». Так на первой итерации мы видим только графический набросок будущего шедевра. Следующий этап грубо, но уверенно добавляет деталей, которые уже на ранних версиях дают представление об изображаемых предметах, композиции и цветовой палитры картины. В конце мы получаем детальное, цветное и полностью реализованное произведение искусства.

Именно технология RAD используется в выбранной среде разработки ПО – Delphi XE10 Berlin под выпуском Embarcadero RAD Studio.

2.2.2 RAD в Delphi XE10 Berlin

Использование методологии RAD в разработке средствами Delphi сводиться к выполнению нескольких простых шагов:

- Расположение визуальных компонентов, предоставляемых стандартными или подключенными библиотеками в нужном месте на визуальной форме (окне программы);
- Установка времени или событий их появления, например, по таймеру или при показе формы;
- Редактирование атрибутов этих элементов: цвет, шрифт, отступы, изображения и других; их событий, таких как «нажатие мыши», «наведение мыши», «нажатие клавиши» и многое другое.

Так во встроенные библиотеки VCL. Delphi входит более 250 визуальных компонентов от форм, простых кнопок, полей ввода и диаграмм, до компонентов связи с базами данных и работы с интернетом.

Главным визуальным компонентом является форма (англ. Form). Именно форма – контейнер для любых элементов в этой среде разработки, представленный в виде «окна» в ОС Windows и необходимый для визуального представления продукта и предоставления пользователю возможности управления этим продуктом.

Вспомогательными элементами служат кнопки, поля ввода, выпадающие списки и многие другие. При разработке достаточно их расположить на форме проекта в необходимых местах и настроить их свойства оформления, такие как высота, ширина, отображаемый текст, его цвет и шрифт и многое другое.

Ко всем элементам проекта, визуальным или нет, можно привязать события и их обработчики, реализующие логику приложения. Наиболее часто использующиеся события:

- OnClick – срабатывает, когда по элементу кликнули курсором мыши;
- OnShow – событие показа элемента, когда элемент находится в поле видимости пользователя;
- OnHide – событие, противоположное OnShow;
- OnCreate – создание элемента, используется в основном при программной реализации элемента в ООП, где необходим контроль памяти;
- OnDestroy – противоположность OnCreate.

Именно элемента взаимодействия с пользователем и их события значительно ускоряют и упрощают работу программисту, однако порой их функционала не хватает. Тогда следует обратиться к фреймворкам, например, популярному детищу российских разработчиков – FireMonkey.

2.2.3 Фреймворк FireMonkey

FMX – российский, разработанный Евгением Крюковым, кроссплатформенный GUI-фреймворк, поддерживающий Windows, Android, Mac OS и Apple iOS, права на который американская компания Embarcadero Technology купила в 2011 году.

FMX совместно с VCL входит в состав Delphi XE2 и старше и использует возможности Pixel Shared 2.0, благодаря которому значительно обогащается графический интерфейс огромным набором визуальных эффектов и позволяет работать с векторными и 3D-интерфейсами.

Основные особенности FireMonkey:

- Единая и, что не менее важно, открытая кодовая база;
- Любой элемент является потенциальным родителем для любого другого;
- Технология LiveBinding позволяет подключать пользовательские типы данных к любому элементу интерфейса;
- Огромная платная и бесплатная база стилей форм и компонентов;
- Технология Multi-Device Preview позволяет настроить визуальное расположение элементов для любой ширины и высоты экранов устройств, на подобии media-запросов в CSS.

Помимо обширного списка особенностей можно перечислить и список возможностей, среди которых необходимо выделить следующее:

- Использование собственного API каждой поддерживаемой платформы;
- Работа со всеми датчиками устройства, такими как GPS, Акселерометр, Компас и другие;
- Push-уведомления;
- Поддержка асинхронных HTTP запросов;

• Работа с огромным количеством БД: MySQL, Oracle, SQLLite, MongoDB и т.д.

Исходя из всего вышеперечисленного видно, что среда программирования Delphi совместно с фреймворком FireMonkey позволяют решить все задачи по разработке бота, зависящих от них. Следующим шагом является организация хранения данных приложения и доступа к ним, другими словами, необходимо выбрать базу данных и систему её управления.

2.4 Система управления базами данных

Одной из важных возможностей ЭВМ является хранение и обработка больших объемов информации, которая реализуется благодаря базам данных.

База данных– набор данных, хранящихся в особом структурированном виде, определенном структурой базы данных. Под данными здесь подразумевается информация, представленная в пригодном для обработки виде.

БД должна не только обеспечивать хранение и обработку данных, но и их безопасность и целостность.

Безопасность данных – их защита от несанкционированного доступа сторонних лиц. Безопасность часто достигается путем защиты данных методом шифрования.

Целостность – возможность восстановления данных в случае их утраты. Целостность особенно важна в случаях, когда с данными могут работать несколько пользователей, так как необходимо сохранить связь данных между собой.

Система управления базами данных – инструмент, позволяющий создавать БД и организовывать ввод данных, их редактирование, обработку и выдачу. СУБД предоставляет средства для выбора интересующих по каким-либо критериям данных и управления ими.

Базы данных по способу установления связи данных делятся на реляционные и иерархические.

Реляционная – наиболее распространенный тип БД, в котором данные представляются в виде таблицы. Главным достоинством такой БД считается простота инструментальных средств её поддержки, недостатком – жесткая структуризация.

Иерархическая БД предполагает наличие связей между данными с общим признаком. Такая связь, как правило, описывается древовидной структурой с односторонней связью от старших вершин к младшим, что ускоряет доступ к необходимым данным.

2.4.1 Реляционная база данных

В разработке бота-редактора имеются точно установленные данные: имеется их тип, их длина и связь друг-с-другом. Исходя из этого нужда в иерархической базе данных отпадает, поэтому фокусируемся на реляционной и её основных аспектах. Одним из основных понятий БД такого типа является «сущность».

Сущность – представление в БД данных о реальных объектах с общими атрибутами, например, студентов высшего учебного заведения. Здесь студент – это сущность, а его атрибутами являются: фамилия, имя, отчество и номер зачетной книжки. Следует понимать, что база данных хранит информацию не об одном студенте, а о десятках, сотнях и более. У каждого из студентов свои значения атрибутов имени, фамилии и так далее.

Атрибут сущности – характеристика или свойство сущности, используемое для её описания. Атрибуты всегда выбираются в зависимости от требований к функционалу разрабатываемой БД. Среди всех атрибутов выделяется особенный, называемый «ключ».

Ключ – атрибут, значение которого уникально для каждой сущности. На практике это могут быть серия и номер паспорта, номера документов, идентифицирующих личность, продукт, услугу. Однако достаточно часто в качестве

ключевого атрибута используют уникальный идентификатор – ID, чья уникальность обеспечивается инкрементами его значения для каждой новой добавленной сущности. Так, например, первый добавленный в БД студент будет иметь ID=1, второй – 2 и каждый n-ый – ID=n, что продемонстрировано на рисунке 6.

Студенты

ID	Фамилия	Имя	Отчество	Номер заче
1	Петров	Виктор	Иванович	1568235
2	Антонова	Ирина	Васильевна	5851684
3	Бронов	Степан	Павлович	5648422

Рисунок 6 – Таблица «Студенты» в СУБД «MS Access»

Идентифицировать конкретную сущность с помощью ключа нужно не только для обращения к ней при редактировании или удалении, но и для её связи с другими таблицами. Так, например, в университете имеется электронный журнал с оценками каждого студента. Целесообразно вывести оценки в отдельную таблицу, в которой, для связи оценок со студентом, будет поле «ID Студента». На практике связь через ключевой атрибут (со значком ключа), будет выглядеть следующим образом, как на рисунке 7:

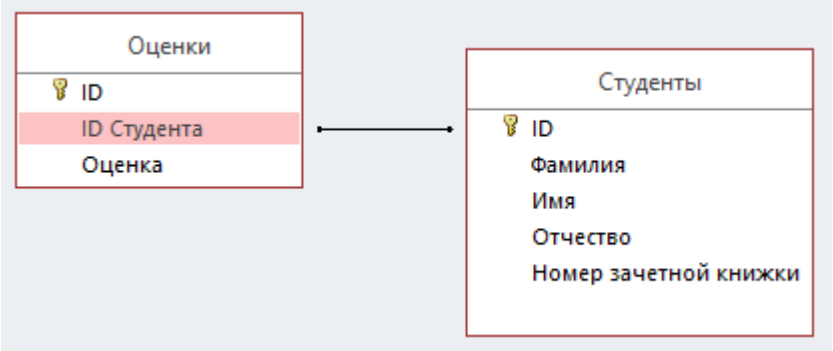

Рисунок 7 – Связь таблиц «Студенты-Оценки»

Следующий вопрос, который необходимо затронуть – это причина, по которой оценки были выведены в отдельную таблицу, ведь можно было создать атрибут «Оценка» для сущности «Студент» в таблице «Студенты».

2.4.2 Нормализация БД

При проектировании баз данных существуют некоторые правила нормализации, созданные с целью исключения избыточного дублирования данных. Это необходимо для устранения аномалий при выполнении основных операций над данным: добавление, обновление и удаление.

Аномалии разделены на следующие категории:

- *Аномалии-модификации* проявляются в том, что при изменении одних данных могут измениться другие записи таблицы;
- *Аномалии-удаления* – при удалении одной записи могут быть удалены данные, не связанные напрямую с удаляемой;
- *Аномалии-добавления* возникают, если не удается добавить неполную запись в БД.

Правила нормализации включают в себя три основные формы.

- *Первая форма нормализации* требует устранения повторяющихся групп в отдельных таблицах, создания отдельных таблиц для каждого набора связанных данных и использование для них ключа. Например, при покупке одного товара у нескольких поставщиков необходимо создать таблицы «Товары», «Поставщики» и абстрактную таблицу связи «Товары-Поставщики», в которой указывать ID_P товара и ID_V поставщика;
- *Вторая нормальная форма* требует создания отдельных таблиц для наборов значений, относящихся к нескольким записям и связи их через внешний ключ. Примером этой НФ является набор оценок для студента (неявка, неуд, удовлетворительно, хорошо, отлично), вынесенный в таблицу «Оценки»,

вместо их выставления в таблице «Студенты». Это позволяет ускорить, упростить и обезопасить процесс проставления оценок тем, что их не нужно каждый раз вводить, реализуя при этом некоторые риски для ИС, зависящие от человеческих факторов таких как опечатка, а достаточно выбрать из определенного набора;

• *Третья нормальная форма* требует удаления поля, не зависящего от ключа. Так, например, при приеме на работу нового сотрудника нужно указать, какой университет он закончил. Это можно сделать путем добавление нового атрибута в таблицу «Сотрудники», однако, если появится нужда изучить этот список университетов для дальнейшего анализа, реализовать это будет крайне проблематично. Поэтому целесообразнее создать таблицу «Университеты» и связать её с «Сотрудники» по соответствующим атрибутам.

Изучив теоретическую информацию о базах данных и системах их управления, а также требования к разрабатываемому ПО из целей и задач необходимо выполнить следующий шаг – выбрать СУБД для конкретного продукта. Выбор пал на SQLite.

2.4.3 СУБД SQLite

SQLite – кроссплатформенная БД, поддерживающая достаточно полный ассортимент SQL-команд и доступна в исходниках на ЯВУ C («си»).

Одной из важных особенностей этой БД, как указано выше, является кроссплатформенность, что в связке с фреймворком FireMonkey позволяет запускать приложение не только на ОС Windows, но и на Linux, Android, и даже на iOS и Mac OS.

Почему «достаточно полный ассортимент SQL-команд»? Сам по себе язык SQL развивается в разные стороны с внедрением различного рода расширений и, хотя и применяются различные стандарты, регулирующие его развитие (например, SQL 92), в реальной жизни многие компании не придержива-

ются стандартов в полной мере. При этом SQLite старается реализовывать минимальный, но полный функционал. Поэтому в этой СУБД есть некоторые неудобства, которые могут быть непривычны многим разработчикам:

- Нет возможности изменить и, уж тем более, удалить столбец из таблицы;
- Триггеры SQLite во многом уступают собратьям из MySQL;
- Не поддерживается UNICODE;
- Нет хранимых процедур;
- Нет технической поддержки в следствии того, что SQLite слишком популярна и настал момент, когда разработчики получали целый вал сообщений о проблемах, многие из которых были из-за невнимательности.

Однако эти минусы меркнут для кроссплатформенных приложений по сравнению с плюсами:

- Такую БД можно легко переносить на другие платформы за счет того, что она представлена в виде одного файла кроссплатформенного формата;
- Тип столбца не определяет тип данных в нем – в любой столбец можно внести значение любого типа. Это похоже на слабую типизацию таких языков программирования, как PHP или JavaScript, где поле типа «INTEGER» может содержать число или строку, например, «2012», которую SQLite попытается привести к целому числу 2012. Тогда для чего вообще нужен тип столбца, если тип значения для него не важен? Тип столбца определяет, как производить сравнение значений – нужно ведь привести значения к единому типу;
- Нет севера. Приложение само является сервером, доступ к которому происходит через подключение при помощи DLL с указанием имени файла. При этом поддерживается несколько подключений к БД одновременно благодаря механизмам блокировки доступа;
- Использование во многопоточных приложениях. SQLite может быть собрана специально для многопоточных приложений с использованием кода

синхронизации и для однопоточных – без этого кода. За множественный доступ отвечает переменная SQLITE_THREADSAFE = 0 по умолчанию. Проверить значение этой переменной можно командой «sqlite3_threadsafe()»

Беря во внимание все её плюсы, при этом не исключая минусы, не трудно догадаться, почему выбор пал именно на эту СУБД: кроссплатформенность, много-поточность и отсутствие сервера.

Следующим шагом после выбора базы данных необходимо выбрать компонент, позволяющий установить связь разрабатываемого ПО через интернет с сервером ВКонтакте для отправки и получения ответов и запросов по защищенному протоколу HTTPS.

2.5 Библиотека Synapse и REST API

Библиотека Synapse была создана с целью обобщения всех классов и функций, отвечающих за работу программирование сети на прикладном уровне с использованием WinSock.

На первый взгляд кажется, что использование стороннего компонента для достаточно тривиальной задачи не оправдано, однако следует учитывать, что большинство стандартных компонентов используют асинхронные методы работы с сетью, а предпочтительнее – синхронные.

Асинхронный метод – подход, когда для подсчета результата функции создается объект, содержащий информацию о том, что результат будет получен или уже пришел. При этом подписка на результат осуществляется через callback. От такого объекта требуется реализовать возможность подписки на результат, сообщать о результате и хранить его.

В синхронном методе результат функции будет передаваться через возвращаемое значение и будет доступен сразу после её выполнения. Это позволяет коду программы выполняться цело и последовательно, чего и нужно добиться, ведь результат (ответ сервиса ВКонтакте) должен быть обработан буквально следующей строкой.

Как уже было сказано выше, Synapse – это набор функций и классов для работы с сетью. Это значит, что, в отличии от концепции RAD и VCL в Delphi, где разработка сводится к перемещению визуальных компонентов, а их установка достаточно время- и трудозатратный процесс, библиотеку, а точнее только необходимые классы, достаточно просто подключить к проекту через специальную директиву «uses».

И так, Synapse упрощает работу с сетью, однако, работая со сторонними сервисами через интернет, разработчик обрекает себя на выбор архитектуры своего ПО. Ведь нужно как-то организовывать взаимодействие компонентов распределенного приложения в сети. В разработке выбор пал на архитектурный стиль REST.

REST – метод представления данных для клиента в удобном формате, что достигается при помощи некоторых согласованных ограничений, правил и REST API, схема работы которого представлена на рисунке 8. Тут клиент, например, приложение на компьютере, через API взаимодействия с серверами через интернет посылает запрос к СУБД. Та в свою очередь его обрабатывает, обращается к базе своих данных и формирует ответ. После чего отсылает его на обработку клиенту, пропустив через API.

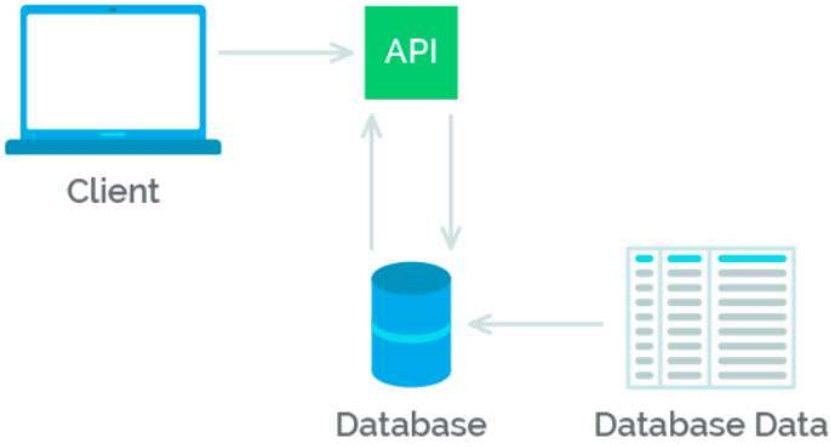

Рисунок 8 – Схема архитектуры REST

REST API реализует четыре базовых операции над данными через методы:

- Получение (GET)
- Создание (POST)
- Обновление (PUT)
- Удаление (DELETE)

По сути, REST API использует единый общий интерфейс для запросов к базам данных, где каждая базовая операция (HTTP метод) соответствует команде в СУБД, что продемонстрировано на рисунке 9, где, например, HTTP метод GET соответствует SQL команде SELECT.

Рисунок 9 – Соответствие HTTP методов SQL командам

Так же, для более простой обработки ответов REST-запросов, каждый запрос имеет статус (или код) ответа, которые классифицируются следующим образом:

- 1XX – статусы ответа информационного характера;
- 2XX – успешные операции;
- 3XX – статусы перенаправления;
- 4XX – ошибки на стороне клиента;
- 5XX – ошибки на стороне сервера.

Так при проверке ответа сервера https://vk.com будет получен статус 200 – сообщающий об успешном запросе к серверу.

Рассмотрим простой пример работы REST API. Для начала необходимо создать некую точку входа, которая существует у любой программы, например, в консольном приложении на C# этой точкой служит функция *main()*, в Delphi - *Application.Initialize*, а в этом примере – файл app.php. Следующим запросом http://site.ru/app.php? **action**=create.user& **name**=Павел& **lastname**=Иванов мы передаем параметрами:

- **action**=create.user – действие по созданию нового пользователя;
- **name**=Павел – имя пользователя
- **lastname**=Иванов – фамилия пользователя

Такого типа взаимодействие приложений в сети очень распространено, поэтому знать, как с ним работать необходимо каждому разработчику. Именно из-за распространённости и поддержки его сервисами ВКонтакте и была выбрана такого рода архитектура приложения. И именно средствами REST API разрабатываемый бот-редактор будет работать с социальной сетью ВКонтакте.

2.6 ВКонтакте API

В этом разделе описана базовая информация о работе с сервисами ВКонтакте средствами ВКонтакте API.

2.6.1 Методы и объекты

API ВКонтакте – это интерфейс взаимодействия с базой данных vk.com через HTTP-запросы, доступный по адресу «https://api.vk.com». Среди особенностей этого предоставляемого функционала можно выделить следующие:

- Разработчику не нужно знать внутреннее устройство БД ВКонтакте (какие таблицы и типы данных используются и какова их связь), достаточно того, что об этом знает API;
- Синтаксис запросов и тип возвращаемых данных строго определены на стороне сервиса.

Попробуем получить информацию о пользователе с id210700286 отправив следующий запрос, представленный на рисунке 10:

```
https://api.vk.com/method/users.get?user_id=210700286&v=5.52
```

Рисунок 10 – Пример запроса информации о пользователе

с параметрами:

- **api.vk.com/method** – адрес сервиса;
- **user.get** – название метода;
- **user_id** – идентификатор пользователя, информацию о котором мы хотим получить;
- **v –** версия используемого API.

В ответе будет получен JSON-объект (данные в виде «ключ»: «значение»), с запрошенными данными представлены на рисунке 11.


```
{"response":[{"id":210700286,"first_name":"Lindsey","last_name":"Stirling"}]}
```


Рисунок 11 – ответ в виде JSON-объекте

2.6.2 Регистрация приложения

Сервис ВКонтакте имеет несколько способов работы со сторонними ресурсами через API. Например, для использования того же метода users.get нам необходимо указать лишь входные параметры, однако для других методов, особенно работающих с загрузкой и выгрузкой информационных ресурсов (фото- или видеоматериалов) этого становится мало. Требуется зарегистрировать свое приложение.

Перед регистрацией и, как следствие, получением заветного API_ID, необходимо выбрать тип приложения. Всего их три варианта:

• Standalone-приложение – это API_ID для мобильных или десктопных приложений-клиентов. Основная идея – запросы осуществляются с персонального устройства пользователя, для которого доступны настройки SDK и push-уведомлений;

• Веб-сайт – API_ID для внешнего сайта, где, к примеру, на PHP работает скрипт с API ВК;

• Flash приложения – приложения, работающие на сервере социальной сети.

После выбора типа приложения вам будет доступен API_ID/APP_ID/client_id, например, 54821452 – число, которое пригодится в дальнейшем.

2.6.3 Авторизация пользователя

Социальная сеть ВКонтакте предоставляет множество информации о пользователях, сообществах и многое другое. При этом часто доступ к информации зависит от того, кто именно пытается её запросить. Например, если вы находитесь в «черном списке» другого пользователя, то вы не сможете отправить ему сообщения т.е. доступ к сообщениям будет закрыт, как и информация о «закрытом профиле».

Так API тоже должен учитывать, кто именно запрашивает доступ к той или иной информации. Именно поэтому большинство методов требуют авторизации.

Для идентификации пользователя используется специальный индивидуальный ключ доступа, называемый access_token. Токен – строка из цифр и латинских символов, передаваемая серверу api.vk.com, с которой он получает информацию. Пример токена представлен ниже на рисунке 12.

```
51eff86578a3bbbcb5c7043a122a69fd04dca057ac821dd7afd7c2d8e35b60172d45a26599c08034cc40a
```

Рисунок 12 – Пример токена

Приобретя ваш собственный токен, вы можете, например, узнать список друзей онлайн через соответствующий запрос на рисунке 13.

```
https://api.vk.com/method/friends.getOnline?v=5.52&access_token=
```

Рисунок 13 – Пример запроса списка друзей

К параметру access_token необходимо добавить свой ключ доступа. В ответе сервер вернет список идентификаторов друзей, находящихся в онлайн.

2.6.4 Права доступа

Социальная сеть ВКонтакте имеет несколько сервисов. Есть сервисы для просмотра, добавления и редактирования видео, аудио и фотографий, сервис, работающий с друзьями или с вашей стеной и так далее. Со всеми сервисами связан определенный набор методов, например, последний запрос «friends.getOnline» обращался к сервису «friends» (друзья), работающему со списком друзей пользователя.

Для каждого сервиса приложение должно иметь определенные права доступа. И часто разрабатываемому ПО не нужны доступы ко всем сервисам, например, приложению для просмотра фотографий нет необходимости работать с сервисом сообщений пользователя.

ВКонтакте для своего приложения можно получить определенный ключ с набором прав для группы сервисов или одного. Этот ключ представляет собой число, являющееся суммой степеней двойки, где каждое слагаемое – права доступа. Например, friends=2, video=4 … group=4096, при этом ключ 6 будет содержать права доступа к «friends» и «video».

Так в главе 2 были определены основные понятия и аспекты разработки бота-редактора и выбраны следующие технологические инструменты:

- Среда программирования Delphi XE10 Berlin;
- Кроссплатформенный фреймворк FireMonkey;
- Система управления базами данных SQLite;
- Библиотека для работы с интернет-запросами – Synapse;
- Функционал для работы с социальной сетью ВКонтакте и её сервисами – API ВКонтакте.

Следующий этап разработки – проектирование и программирование бота.

3 Программная реализация

В главе описан процесс реализации бота-редактора для групп социальной сети ВКонтакте. А точнее анализ требований, процесс разработки базы данных и разработка самого приложения, работающего с социальной сетью.

3.1 Разработка БД

Разработка базы данных достаточно трудоемкий и сложный процесс, требующий максимального погружения и понимания целей и задач разрабатываемого ПО.

Так, исходя из функциональных требований к программе, необходимо реализовать хранение:

- Пользовательских данных и данных профилей;
- Информацию о задачах, выполняемых программой.

Пользовательские данные и данные профилей представляют собой совокупность атрибутов, необходимых для идентификации человека сервисами ВКонтакте, а также для предоставления необходимого доступа к ним.

Необходимыми и достаточными для работы бота-редактора данными являются:

- UserID – идентификатор пользователя в социальной сети;
- Login и Password – текстовые поля, содержащие информацию, необходимую для авторизации пользователя;
- Token – текстовое поле, содержащее ключи доступа профиля к сервисам ВКонтакте;
- Name и LastName – поля с именем и фамилией пользователя, полученные средствами социальной сети при успешной авторизации.

На рисунке 14 представлена таблица для хранения этой информации в БД.

Рисунок 14 – Таблица хранения пользовательской информации

Информация о задачах программы – перечень атрибутов, необходимых и достаточных для того, чтобы бот-редактор мог бесперебойно и корректно выполнять свою работу. Среди таких атрибутов были выявлены следующие:

- TaskID – числовой идентификатор задачи в списке задач;
- WallGetFrom – поле, содержащее перечень сообществ-доноров, откуда избираются посты для анализа и публикации;
- WallAdmisTypes – информация о допустимых типах постов, которые следует рассматривать боту;
- WallGetCount – количество постов из одного сообщества, выбирающихся для анализа за один цикл работы бота;
- WallInvalidWords – перечень слов, при нахождении которых в посту, бот его игнорирует. Благодаря этому приложение способно отсеять рекламные посты или посты, содержащие нежелательный контент;
- WallPostTo – идентификатор сообщества, в которое ПО публикует посты;
- WallPostFrom – идентификатор пользователя, от имени которого будет происходить публикация поста;
- TimeOffset – время в минутах, через которое начнется новый цикл работы программы;
- Status – статус бота, изменяющийся во время выполнения работы;
- TaskName – имя задачи, установленное пользователем бота;

• WorkUserID – идентификатор пользователя, через которого будет работать бот с сервисами ВКонтакте.

Таблица хранения данных о задачах представлена на рисунке 15 ниже.

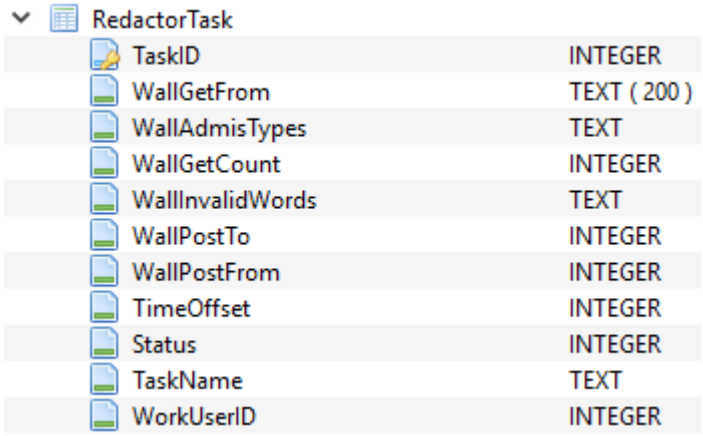

Рисунок 15 – Таблица хранения информации о задачах бота

Проанализировав требования к разрабатываемому ПО была спроектирована и разработана база данных для хранения основных данных бота. Следующим этапом является разработка приложения бота-редактора.

3.2 Разработка бота-редактора

Одной из основных идей разработки ПО стала идея модульности конечного продукта, то есть разбиение требуемого функционала на независимые модули, работающие друг-с-другом.

Исходя из требований было принято решение разбить программу на следующие составляющие:

- «Загрузчик»;
- «Улей»;
- «Задачник»;
- «Историк»;
- «Коннектор»;

• «Шеф».

«Загрузчик» - не визуальный модуль, отвечающий за загрузку и выгрузку данных о профилях и задачах бота во время его запуска. Для удобства и индикации загрузки в приложении на главном экране имеется стандартный для ОС Windows 8, 8.1 и 10 элемент загрузки в виде анимированного круга, как на рисунке 16.

Рисунок 16 – Индикатор загрузки для модуля «Загрузчик»

Во время загрузки и инициализации всех составляющих компонентов бота, пользователь не имеет доступ к управлению, иначе говоря, интерфейс программы никак не взаимодействует с ним, а находится в режиме ожидания. После загрузки данных, рабочая область становится активна и позволяет пользователю работать с ботом, при этом вне зависимости от успешности загрузки, приложение выведет информацию в разделе «Логи».

«Улей» - модуль бота-редактора, отвечающий за работу с профилями в системе. В его задачи входит проверка актуальности профилей через запрос к сервису ВКонтакте, добавление новых и редактирование старых профилей.

Для пользователя этот модуль представлен в виде списка подключенных профилей как на рисунке 17.

Для модуля «Рабочий» были разработаны классы по работе с профилями и их данными, представленные на рисунке 18:

TProfile – класс по работе с одним профилем. Он хранит информацию о профиле: пароль, логин, имя, фамилия, его идентификатор во ВКонтакте. Кроме хранения данных класс предоставляет список базовых операций над профилем, такие как получение информации о нем от сервисов социальной сети, попытка авторизации;

TProfileList – класс-хранилище всех подключенных профилей. Именно он отвечает за многопользовательность приложения.

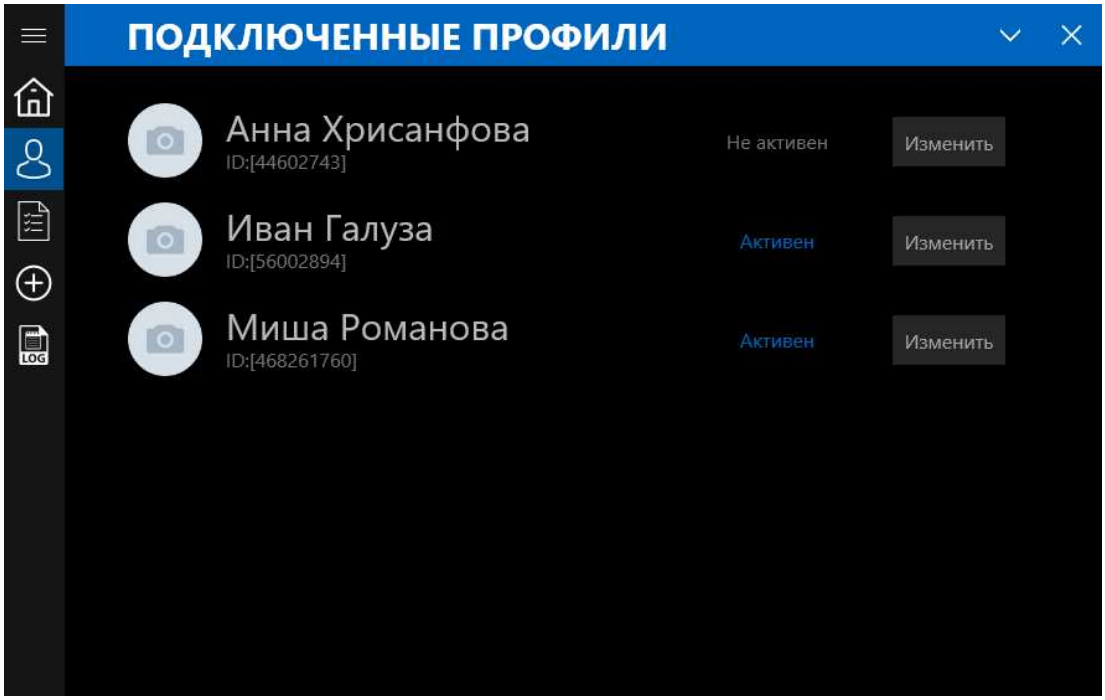

Рисунок 17 – Список профилей в модуле «Рабочий»

Рисунок 18 – Классы по работе с профилями

«Задачник» - модуль, отвечающий за задачи, которые выполняет бот во время своей работы. Модуль выводит пользователю список задач, рабочий профиль, прикрепленный к каждой из них, индикатор её выполнения и текущий статус. Все это представлено на рисунке 19 ниже.

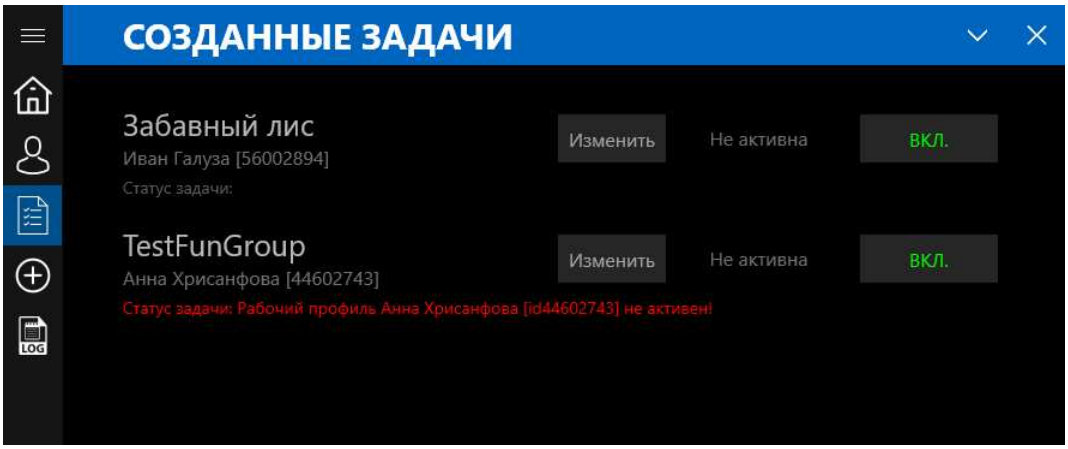

Рисунок 19 – Модуль «Задачник» со списком существующих задач

Для этого модуля были разработаны классы TBTask и TRedactorTask которые реализуют саму задачу и процессы над ней и продемонстрированы на рисунке 20 и 21 соответственно.

Рисунок 20 – Класс TBTask

TBTask - базовый класс с основным функционалом задач для наследуемого TRedactorTask. Здесь указаны идентификатор задачи (TaskID), её наименование в программе (TaskName), рабочий профиль (WorkUserID) и метод, управляющий статусом задачи (SetStatusDescript);

TRedactorTask – класс, базирующийся на TBTask и содержащий в себе весь функционал задач конкретно для бота-редактора:

- Время простоя задачи (TimeOffset);
- Количество постов, обрабатываемых за один цикл (WallGetCount);
- Идентификатор групп, представленных для анализа (WallGetFrom);
- Список недопустимых символов (WallInvalidWords);
- Получение (GetAdmisTypes) и установка (SetAdmisTypes) типов постов для обработки.

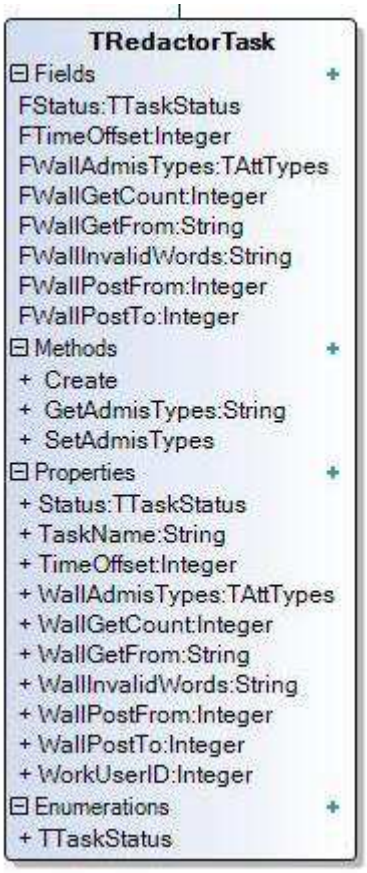

Рисунок 21 – Класс «TRedactorTask»

«Историк» - чуть ли не самый важный компонент программы, фиксирующий, сохраняющий и представляющий в удобном для чтения виде все действия всех модулей в программе для анализа разработчиком или пользователем. Представляет отчетность в виде списка действий бота в хронологическом порядке и подсветкой разного цвета, зависящего от важности записи или модуля, выполнившего действие. Такая отчетность представлена на рисунке 22.

Как и для всех предыдущих модулей, для «Историка» также разработаны отдельные классы работы над сообщениями бота, представленные на рисунке 23:

TMessanger – хранилище модулей, сообщения которых принимает «Историк»;

TMsgClass – класс, реализующий прослушивание сообщений бота и их визуализацией;

TLogMess – хранилище типов сообщений.

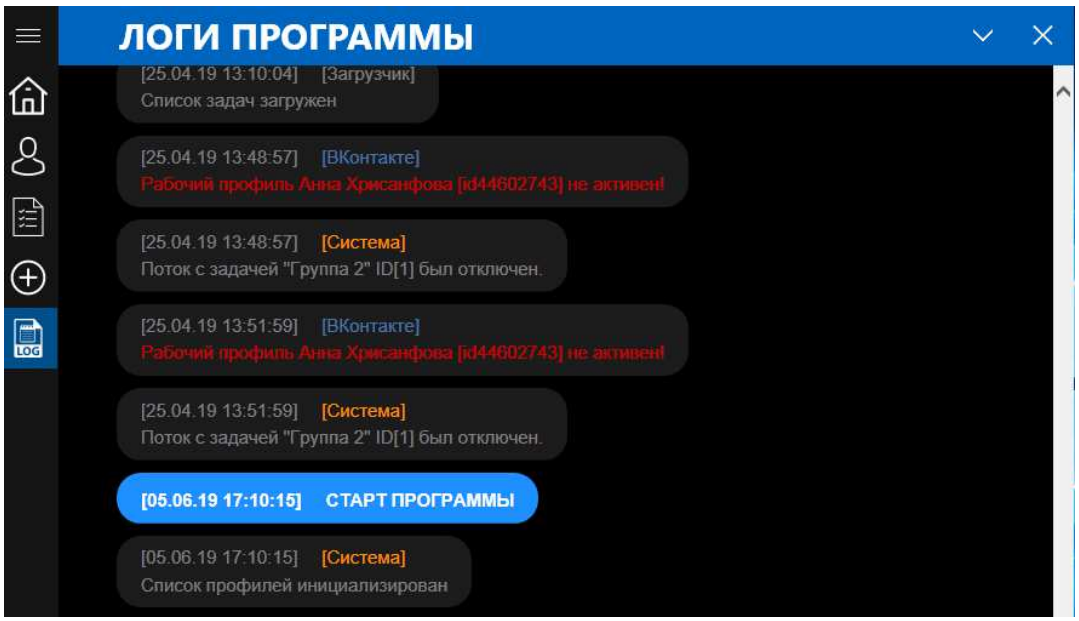

Рисунок 22 – Отчетность модуля «Историк»

Рисунок 23 – Классы модуля «Историк»

Как видно на рисунке 22, основными цветами являются:

• Голубой – разделительный блок начала и конца работы бота;

• Синий – цвет подмодуля «Система проверок»;

• Темно-синий – официальный цвет ВКонтакте, именно поэтому им обозначены все работы с сервисами этой социальной сети;

• Оранжевый – цвет работы системы по регулированию всех процессов взаимодействия модулей;

• Серый – цвет модуля «Загрузчик»;

• Красный – ошибка или важное предупреждение, которое не стоит упускать из виду.

«Коннектор» - модуль, отвечающий за работу бота с сервисами ВКонтакте через интернет. Модуль содержит библиотеку Synapse и методы отправки и получения данных через интернет-протокол HTTPS и служит посредником между программой и социальной сетью, позволяющем абстрагироваться программисту от этого взаимодействия. Схематически такая абстракция представлена на рисунке 24, где бот-редактор на высоком уровне через «Коннектор» (VKAPI) взаимодействует с ВКонтакте API на более низком уровне.

Рисунок 24 – Абстракция взаимодействия бота с сервисами ВКонтакте

Для работы с социальной сетью здесь описаны следующие методы:

- GetVKError – обработка ошибок ВКонтакте;
- WallPost – позволяет создать запись на стене;
- WallGet – возвращает список записей со стены сообщества или пользователя;
- UsersGet – возвращает расширенную информацию о пользователях;
- PhotosCopy – позволяет скопировать фотографию в альбом «Сохраненные фотографии»;
- PhotosDelete – удаление фотографий на сайте.

«Шеф» - модуль, реализующий многопоточность, т.е. позволяет приложению одновременно выполнят несколько задач.

Такое название модуль имеет не просто так, ведь его работа очень похожа на работу шефа на кухне – он следит за каждым подчиненным (потоком): дает или отменяет ему заказ на определенное блюдо, отправляет его в отпуск или срочно вызывает на рабочее место и так далее. Схематично такая работа изображена на рисунке 25, где «*» отмечены указатели на потоки из их хранилища.

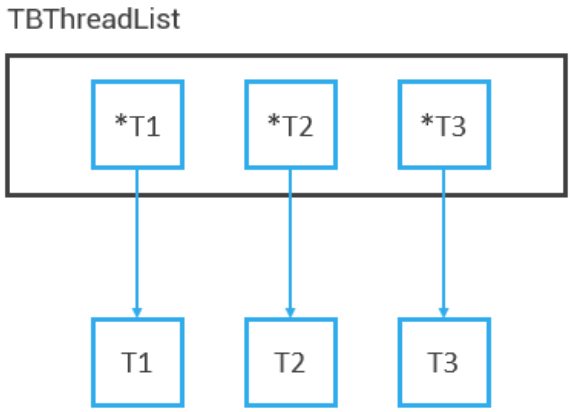

Рисунок 25 – Схема контроля потоков в модуле «Шеф»

Для того, чтобы это все работало должным образом, была реализована иерархия классов потоков и их хранилищ.

TBThread – базовый класс потока, представленный на рисунке 26, включающий в себя:

- Create - метод создания потока,
- Stop – метод остановки;
- AfterTerminate – событие уничтожения и описание того, что будет происходить перед ним;
- TaskID – идентификатор задачи, которую он должен выполнять во время своей работы.

Рисунок 26 – Базовый класс потока TBThread

TBThreadList – базовый класс хранилища потоков представленный на рисунке 27 и имеющий следующее описание:

- Create и Destroy – методы создания и удаления хранилища;
- StartAll и StopAll – запуск и остановка всех потоков в хранилище;
- StartByTaskID и StopByTaskID – запуск и остановка конкретного потока по его TaskID.

Рисунок 27 – Базовый класс хранилища потоков TBThreadList

TRedThread – класс, реализующий выполнение конкретной задачи для бота-редактора, в теле метода Execute которого описана логика обработки задачи.

TRedThreadList – класс-хранилище для задач, исполняемых программой, в наследуемом методе StartByTaskID которого описана логика начала исполнения задачи редактора.

Так классы TRedThread, TRedThreadList со своими базовыми классами изображены на рисунке 28.

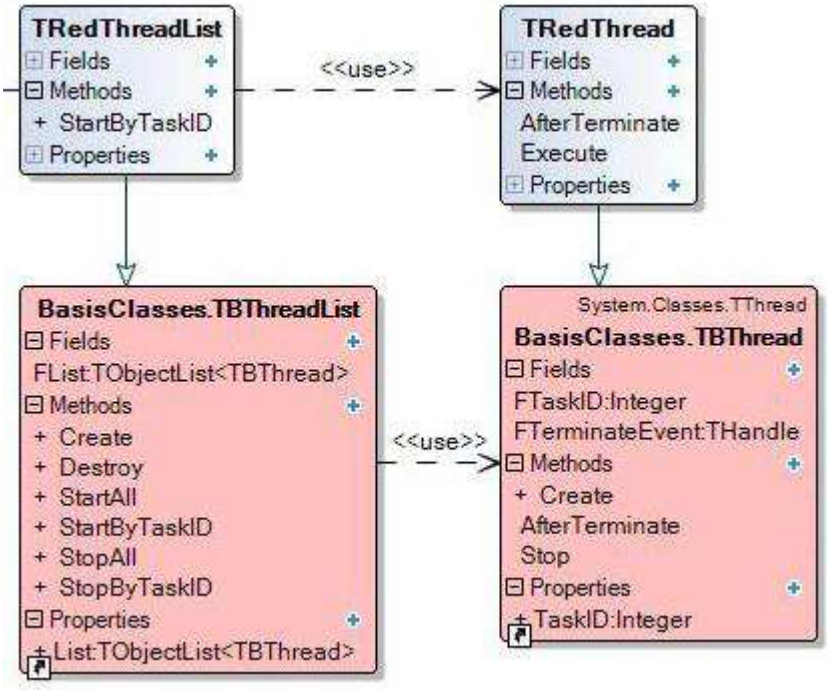

Рисунок 28 – Родительские и дочерние классы потоков и их хранилищ

ЗАКЛЮЧЕНИЕ

В процессе выполнения выпускной квалификационной работы был разработан бот-редактор для групп ВКонтакте, оптимизирующий и автоматизирующий работу специалиста по подбору и публикации контента (контент-менеджера). Данное ПО является легким и качественным маркетинговым инструментом для работы с самой разной аудиторией сообщества, созданный для его популяризации.

При разработке были решены следующие задачи:

- Работа с несколькими пользователями;
- Параллельная работа с несколькими задачами;
- Автоматизация процесса поиска и публикации информационного материала;
- Обеспечение работы с сервисами ВКонтакте через интернет;
- Тестирование приложения.

Таким образом была достигнута основная цель – автоматизация работы контент-менеджера сообществ для минимизации расходов и увеличения производительности, приводившей к увеличению популярности групп.

СПИСОК СОКРАЩЕНИЙ

API - Application Programming Interface - интерфейс программирования приложений;

BMW - Bayerische Motoren Werke - Баварские моторные заводы;

CSS - Cascading Style Sheets - каскадные таблицы стилей;

DLL - Dynamic Link Library - динамически подключаемая библиотека;

FMX – FireMonkey;

GPS - Global Positioning System - система глобального позиционирования;

GUI - Graphical User Interface - Графический интерфейс пользователя;

HTTP - HyperText Transfer Protocol - протокол передачи гипертекста;

HTTPS - HyperText Transfer Protocol Secure – защищённый протокол передачи гипертекста;

JSON - JavaScript Object Notation - Нотация объектов JavaScript;

PHP - Hypertext Preprocessor - препроцессор гипертекста;

RAD - Rapid Application Development — быстрая разработка приложений;

REST - Representational State Transfer - передача состояния представления;

SEO - Search Engine Optimization - поисковая оптимизация;

SQL – Structured query language – язык структурированных запросов;

VCL – Visual Component Library – библиотека визуальных компонентов;

БД – база данных;

ЖЦ – жизненный цикл;

ИС – информационная система;

НФ – нормальная форма;

ОС – операционная система;

ПО – программное обеспечение;

РФ – Российская Федерация;

СУБД – система управления базами данных;

ЯВУ – язык высокого уровня;

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. СТАНДАРТ ОРГАНИЗАЦИИ Система менеджмента качества. Общие требования к построению, изложению и оформлению, документов учебной деятельности СТО 4.2-07-2014 – Красноярск: ИПК СФУ, 2014. – 60 с.

2. Что такое «лайки» и зачем они нужны? [Электронный ресурс] / КакБог.ru – Саморазвитие Самосовершенствование Личностный рост. – Режим доступа: http://kak-bog.ru/chto-takoe-layki-i-zachem-oni-nuzhny

3. Боты в соцсетях: что это такое, и с чем их едят? [Электронный ресурс] / Gagadget.com – нескучный сайт о технике. – Режим доступа: https://ga-gadget.com/how-it-works/20734-botyi-v-sotssetyah-chto-eto-takoe-i-s-chem-ih-edyat/

4. Социальное ботоводство: кто, как и зачем использует ботов? [Электронный ресурс] / Хабр. – Режим доступа: https://habr.com/ru/post/322706/

5. Что такое бот в Интернете, и чем он полезен пользователям? [Электронный ресурс] / НаВолне – поймай свою волну вместе с нами! – Режим доступа: https://www.navolne.life/post/chto-takoe-bot-v-internete-i-chem-on-polezen-pol-zovatelyam

6. Что такое бот и зачем он нужен? [Электронный ресурс] / КакБог.ru – Саморазвитие Самосовершенствование Личностный рост. – Режим доступа: http://kak-bog.ru/chto-takoe-bot-i-dlya-chego-nuzhen

7. Кто такой контент менеджер, чем он занимается и как им начать с нуля [Электронный ресурс] / IDEA37.INFO – Прочь из офисов, к себе настоящему через саморазвитие и путешествия. – Режим доступа: https://idea37.info/stati/kto-takoj-kontent-menedzher/

8. Профессия контент-менеджер: обязанности, требования, зарплата [Электронный ресурс] / ИМ – блог про интернет-маркетинг. – Режим доступа: https://internet-marketings.ru/professiya-kontent-menedzher/

9. Что такое сообщества ВКонтакте [Электронный ресурс] / Semantica. – Режим доступа: https://semantica.in/blog/chto-takoe-soobshhestva-vkontakte.html

10. Что такое сообщество? Сообщества в «ВКонтакте». Мировые сообщества [Электронный ресурс] / ФБ.ру. – Режим доступа: http://fb.ru/article/155307/chto-takoe-soobschestvo-soobschestva-v-vkontakte-mirovyie-soobschestva

11. RAD (программирование) [Электронный ресурс] / ВикипедиЯ – Свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/RAD_(программирование)

12. Что же такое FireMonkey? [Электронный ресурс] / Softacom – разработка программного обеспечения в области систем безопасности. – Режим доступа: http://www.softacom.ru/ru_chto_zhe_takoe_firemonkey

13. FireMonkey [Электронный ресурс] / ВикипедиЯ – Свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/FireMonkey

14. Разработка мобильных приложений в Embarcadero FireMonkey (FMX 6) FireMonkey [Электронный ресурс] / Хабр. – Режим доступа: https://habr.com/ru/post/222307/

15. Руководство по проектированию реляционных баз данных (10-13 часть из 15) [перевод] FireMonkey [Электронный ресурс] / Хабр. – Режим доступа: https://habr.com/ru/post/193756/

16. Нормализация базы данных и её формы FireMonkey [Электронный ресурс] / Office-menu – Статьи и уроки Excel и SQL. – Режим доступа: http://office-menu.ru/uroki-sql/51-normalizatsiya-bazy-dannykh

17. Описание основных приёмов нормализации базы данных Synapse [Электронный ресурс] / Служба поддержки Майкрософт. – Режим доступа: https://support.microsoft.com/ru-ru/help/283878.

18. SQLite – замечательная встраиваемая БД (часть 1) FireMonkey [Электронный ресурс] / Хабр. – Режим доступа: https://habr.com/ru/post/149356/

19. Введение в REST API [Электронный ресурс] / <MyRusakov.ru/>. – Режим доступа: https://myrusakov.ru/rest-api-introduction.html

20. Введение в Synapse API [Электронный ресурс] / Delphi.int.ru – порт@л программистов. – Режим доступа: http://www.delphi.int.ru/articles/49/

21. Библиотека Synapse [Электронный ресурс] / DelphiComponent.ru – бесплатно видеоуроки по Delphi, статьи, исходники. – Режим доступа - https://delphicomponent.ru/317-biblioteka-synapse.html

22. Работа с библиотекой Synapse в Delphi – начало [Электронный ресурс] / Парсинг от А до Я – блог о программировании парсеров и web-автоматизации. – Режим доступа – http://parsing-and-i.blogspot.com/2010/02/synapse-in-delphi-first-steps.html

23. Правильный подход к использованию API Вконтакте [Электронный ресурс] / Хабр. – Режим доступа: https://habr.com/ru/post/221949/

24. Асинхронные API и объект Deferred в деталях Вконтакте [Электронный ресурс] / Хабр. – Режим доступа - https://habr.com/ru/post/175947/

25. Что нам даёт API ВКонтакте Вконтакте [Электронный ресурс] / Flash-разработка. – Режим доступа: http://racer242.blogspot.com/2010/01/api.html

26. Знакомство с API ВКонтакте Synapse [Электронный ресурс] / VK Developers. – Режим доступа: https://vk.com/dev/first_guide

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
Информационные системы

УТВЕРЖДАЮ
Зав. кафедрой ИС
_____ П.П. Дьячук
подпись инициалы, фамилия
«21» 06 2019г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 Информационные системы и технологии

Разработка бота-редактора для групп ВКонтакте

Руководитель	_____ 21.06.19 подпись, дата	доцент, к.т.н.	И.А. Легалов
Выпускник	_____ 21.06.19 подпись, дата		И.В. Галуза
Нормоконтролер	_____ 21.06.19 подпись, дата	ст. преподаватель	Ю. В. Шмагрис

Красноярск 2019