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

Институт космических и информационных технологий
Кафедра информатики

УТВЕРЖДАЮ
Заведующий кафедрой
_____ А.И. Рубан
подпись инициалы, фамилия

«_____» _____ 2016 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

231000.62 – Программная инженерия
код – наименование направления

Разработка облачного web-приложения для сбора баз ретаргетинга

Руководитель _____ _____ В.А. Красиков
подпись, дата должность, ученая степень инициалы, фамилия

Выпускник _____ В. Ф. Непомнящих
подпись, дата инициалы, фамилия

Нормоконтроль _____ _____ О.А. Антамошкин
подпись, дата должность, ученая степень инициалы, фамилия

Красноярск 2016

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ КОСМИЧЕСКИХ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
Кафедра Информатики

УТВЕРЖДАЮ
Заведующий кафедрой
_____ А.И. Рубан
« _____» _____ 2016 г. З

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы

Студенту: Непомнящих Вячеславу Федоровичу

Группа: КИ12-18Б

Направление (специальность): 231000.62

Наименование: «Программная инженерия»

Тема выпускной квалификационной работы: «Разработка облачного web-приложения для сбора баз ретаргетинга»

Утверждена приказом по университету: № 6145/с от 10.05.2016

Руководитель ВКР: В.А. Красиков

Исходные данные для ВКР: анализ спроса и требований к системам подобного типа

Перечень разделов ВКР: Описание предметной области, техническое задание, описание и обоснование выбора инструмента для разработки программного обеспечения, моделирование и проектирование программного обеспечения, описание разработанного программного обеспечения.

Перечень графического материала:

Руководитель ВКР _____

Задание принял к исполнению _____

« ___ » _____ 2016г.

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка облачного web-приложения для сбора баз ретаргетинга» содержит 50 страниц текстового документа, 10 использованных источников, 16 иллюстраций.

РЕКЛАМА, СОЦИАЛЬНЫЕ СЕТИ, РЕТАРГЕТИНГ, ОБЛАЧНОЕ ПРИЛОЖЕНИЕ, АНАЛИЗ ТРЕБОВАНИЙ, ТЕХНИЧЕСКОЕ ЗАДАНИЕ, ПРОЕКТИРОВАНИЕ, РАЗРАБОТКА, ПРИЛОЖЕНИЕ.

Целью работы является разработка программного обеспечения для автоматизации процесса поиска целевой аудитории в социальной сети «ВКонтакте».

Приложение предназначено для интернет-маркетологов, трафик-менеджеров, smm-специалистов и собственников малого бизнеса.

Разработанное программное обеспечение расширяет стандартные возможности по настройке таргетированной рекламы социальной сети, и позволяет упросить процесс поиска целевой аудитории, сделать его более гибким, и в результате, значительно, уменьшить стоимость целевого действия.

Исходя из поставленной цели, в работе приведены: общий анализ требований к подобным системам, обзор существующих приложений на рынке и их сравнение, анализ технологий необходимых для реализации приложения, техническое задание, диаграммы вариантов использования, описание базы данных и описание разработанного приложения.

В результате работы разработано web-приложение, автоматизирующее процесс сбора баз ретаргетинга, которое написано с использованием фреймворка Ruby on Rails на языке программирования Ruby с использованием HTML, CSS и JS.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ **3**

Цели и задачи 4

1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ **5**

1.1 Введение в предметную область 5

1.2 Требования, предъявляемые к подобным системам 7

1.3 Обзор существующих решений 8

1.3.1 Анализ приложения «Церебро» 9

1.3.2 Анализ приложения «Сегменто-Таргет» 10

1.3.3 Анализ приложения «Retarget.sexy» 11

1.3.4 Выводы 12

2 ТЕХНИЧЕСКОЕ ЗАДАНИЕ **13**

2.1 Наименование и основание для создания 13

2.1.1 Полное наименование 13

2.1.2 Основание для создания 13

2.2. Назначение и цели создания системы 13

2.2.1 Назначение разрабатываемой системы 13

2.2.2 Цели создания системы 13

2.3. Требования к системе 14

2.4. Требования к структуре и функционированию системы 15

2.4.1 Перечень модулей, их назначение и основные характеристики 15

2.5. Требования к модулям системы 16

2.5.1 Требования к модулю «Аутентификация» 16

2.5.2 Требования к модулю «Задания» 16

2.5.3 Требования к модулю «Отчеты» 17

2.5.4 Требования к модулю «Оплата» 18

2.6. Требование к архитектуре системы 19

2.7. Требования к интерфейсу системы 20

2.8. Требования к надежности системы 20

2.9. Требования к техническому обеспечению 21

3 ОПИСАНИЕ И ОБОСНОВАНИЕ ВЫБОРА ИНСТРУМЕНТА ДЛЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ **22**

3.1 Требования к инструментам серверной части приложения 22

3.1.1 Система управления базами данных 22

3.1.1.1 Обзор современных СУБД 23

3.1.1.1.1 Firebird 23
3.1.1.1.2 MySQL 23
3.1.1.1.3 PostgreSQL 24
3.1.1.2 Выбор СУБД 25
3.1.2 Языки программирования 26
3.1.3 Веб–сервер 27
3.1.3.1 Обзор современных веб– серверов 27
3.1.3.1.1 Обзор веб–сервера Ruby on Rails + Apache 27
3.1.3.1.2 Обзор веб–сервера Ruby on Rails + Nginx + Unicorn 27
3.1.3.1.3 Обзор веб–сервера Ruby on Rails + WEBrick 30
3.1.3.2 Выбор веб–сервера 30
3.2 Требования к клиентской части приложения 31
4 МОДЕЛИРОВАНИЕ И ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 31
4.1 Архитектура проектируемого приложения 31
4.2 Построение диаграммы вариантов использования 33
4.3 Описание информационной базы 36
4.5. Вывод 38
5 ОПИСАНИЕ РАЗРАБОТАННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 39
5.1 Модуль «Аутентификация» 39
5.2 Панель управления 41
5.3 Модуль «Задания» 42
5.4 Модуль «Отчеты» 42
5.5 Модуль «Оплата» 43
ЗАКЛЮЧЕНИЕ 44
СПИСОК СОКРАЩЕНИЙ 45
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 46

ВВЕДЕНИЕ

Интернет – единственный сектор рекламной отрасли, показавший рост в 2015 году. На фоне общего спада экономики и сокращения рекламных бюджетов маркетологи, специалисты по рекламе, трафик-менеджеры, собственники малого бизнеса ищут самые оптимальные способы и площадки для проведения рекламных компаний. Социальные сети – это одна из самых актуальных площадок для рекламных кампаний в 2015-2018 годах. Вопреки общему мнению о том, что в социальные сети люди приходят за общением и развлечениями, именно в социальных сетях есть возможность наиболее эффективно контактировать со своей целевой аудиторией. На примере: классический баннер, расположенный около дороги «показывается» всем подряд, отследить его эффективность в краткосрочной перспективе просто невозможно. Ведь мы даже не знаем, кто и когда видит этот баннер. Напротив, реклама в социальных сетях позволяет на основе публичных данных о пользователях таких как: ФИО, город, пол, возраст, увлечения, жизненная позиция и даже список любимых сериалов, составлять максимально эффективные рекламные сообщения для каждой группы целевой аудитории.

Разрабатываемое ПО, предлагает еще более эффективный инструмент для поиска целевой аудитории, основываясь не только на тривиальных данных, которые указаны самими пользователи, но и учитывающие их поведенческие характеристики и публичную активность. Обладая такой широкой информацией о своей аудитории, ее можно сегментировать еще «уже» и предлагать для каждой подгруппы свое уникальное рекламное сообщение, тем самым повышая эффективность рекламной кампании за счет снижения стоимости «контакта» с целевой аудиторией и снижения стоимости целевого действия.

Цели и задачи

Цель: предоставить удобный, быстрый и недорогой сервис для маркетологов, smm-специалистов, трафик-менеджеров, специалистов по рекламе и собственников малого бизнеса, позволяющий находить и сегментировать свою целевую аудиторию в социальной сети «ВКонтакте», в целях оптимизации проводимых рекламных кампаний путем составления более релевантных рекламных сообщений для каждой из подгрупп целевой аудитории.

Задачи: разработать приложение с использованием фрейворка Ruby on Rails отвечающее запросам целевой аудитории потенциальных пользователей, с удобным интерфейсом и небольшой стоимостью доступа к приложению.

1 Описание предметной области

1.1 Введение в предметную область

Предметной областью данной выпускной квалификационной работы является работа веб-приложения для сбора баз ретаргетинга. Стандартные возможности по настройке таргетированной рекламы в социальной сети позволяют оперировать лишь небольшим количеством данных, такими как:

- ФИО;
- Возраст;
- Политические взгляды;
- Религиозные убеждения;
- Любимые книги, фильмы или музыка;
- Интересы, например: путешествия.

Как правило, эти данные пользователь указывает при регистрации и со временем они устаревают. Например, в 2009 году пользователь социальной сети «ВКонтакте» учился в 11 классе и очень любил фильмы о Гарри Поттере. Именно эти данные он и указал при регистрации в социальной сети. С тех пор прошло 7 лет, пользователь повзрослел и теперь интересуется не фильмами о Гарри Поттере, а автомобилями и бизнесом. Показывать такому пользователю рекламу, например, нового коллекционного сборника книг нерационально, пользователь уже интересуется другими вещами. Безусловно, есть определенная вероятность того, что какая-то часть таких пользователей все еще интересуется тем, о чем они ранее писали в своем профиле. Именно так работала реклама «ВКонтакте» до появления возможности ретаргентинга, т.е показа рекламы пользователям, которые уже совершили какое-то целевое действие. Например, упомянутый выше пользователь, в 2016 году активно интересуется машинами и бизнесом, а это значит, что он проявляет активность в тех сообществах, где публикуется информация о бизнесе и машинах.

Приложения для сбора баз ретаргентинга позволяют находить сообщества, по ключевым словам, например, «автомобили» и «бизнес», анализировать контент, представленный в этих сообществах и находить тех пользователей, которые проявляют там активность путем постановки отметок «мне нравится», «рассказать друзьям» и написанием комментариев.

Показывать рекламу автомобилей людям, которые ими интересуются гораздо выгоднее, чем показывать эту же рекламу всем подряд. В этом и заключается основное преимущество таких приложений – они позволяют значительно уменьшить стоимость целевого действия за счет подбора более релевантной аудитории.

Глоссарий предметной области представлен в таблице 1.

Таблица 1 – Глоссарий предметной области

Термин	Определение
Сообщества «ВКонтакте»	Группа людей, объединённая общими интересами, которые собираются в специальных разделах социальной сети, которые называются «группы» и «публичные страницы»
Таргетинг	Возможность показывать рекламные объявления пользователям на основе данных о них
Ретаргетинг	Возможность показывать рекламные объявления пользователям, которые проявили некую заинтересованность в продвигаемом продукте
Целевое действие	Действие, совершенное пользователем, которое ожидает рекламодатель. Например: покупка, заявка, звонок, подписка на email-рассылку и так далее.

1.2 Требования, предъявляемые к подобным системам

На основании анализа существующих систем, а также на основании опроса пользователей был составлен общий перечень требований к разрабатываемому ПО:

- Высокая скорость работы;
- Вычисления на стороне сервера, а не клиента;
- Удобный интерфейс;
- Невысокая цена (до 1000р/мес.).

Из дополнительных требований, которые были выявлены в результате более подробного опроса пользователей можно отметить следующее:

- Наличие оперативной технической поддержки;
- Наличие обучающих материалов;

1.3 Обзор существующих решений

На текущий момент на рынке подобных систем предоставлено множество игроков. Приведем исследование трех наиболее популярных приложений: Церебро, Сегменто-Таргет и Retarget.sexy. Анализ будет производиться по следующим параметрам:

- Cost – цена за доступ к сервису;
- Speed – скорость работы;
- UI – удобство пользовательского интерфейса;
- ТП – наличие технической поддержки;
- Test – наличие тестового периода;
- Popular – известность на рынке;
- Features – прочие особенности.

Сравнение существующих продуктов приведено в таблице 2.

Name	Cost	Speed	UI	ТП	Test	Popular	Features
Церебро	700р/мес	4/10	6/10	+	-	9/10	Нет облачной версии
Сегменто-Таргет	От 200 до 1500р/мес	6/10	3/10	-	+	6/10	Облачная версия стоит дороже
Retarget.sexy	499р/мес	8/10	8/10	-	+	3/10	Полностью облачная версия

Таблица 2 – Анализ существующих систем.

1.3.1 Анализ приложения «Церебро»

Самый популярный аналог «Церебро» имеет развитый функционал, хорошую техническую поддержку, базу обучающих материалов, но достаточно неудобный интерфейс и низкую скорость работы. При обработке больших массивов данных, отсутствие облачной версии приложения и низкая скорость работы, становятся большим минусом приложения. Все вычисления производятся на стороне «клиента», что делает работу с сервисом для владельцев не передовой компьютерной техники просто невозможной. В процессе тестирования приложение показывало нестабильную работу при работе с объемом около 100 тысяч записей. Несмотря на то, что выборка такого объема удовлетворит запросы более половины потенциальных пользователей, вторая половина аудитории просто не сможет удовлетворить свои запросы. На рисунке 1 представлен интерфейс приложения «Церебро».

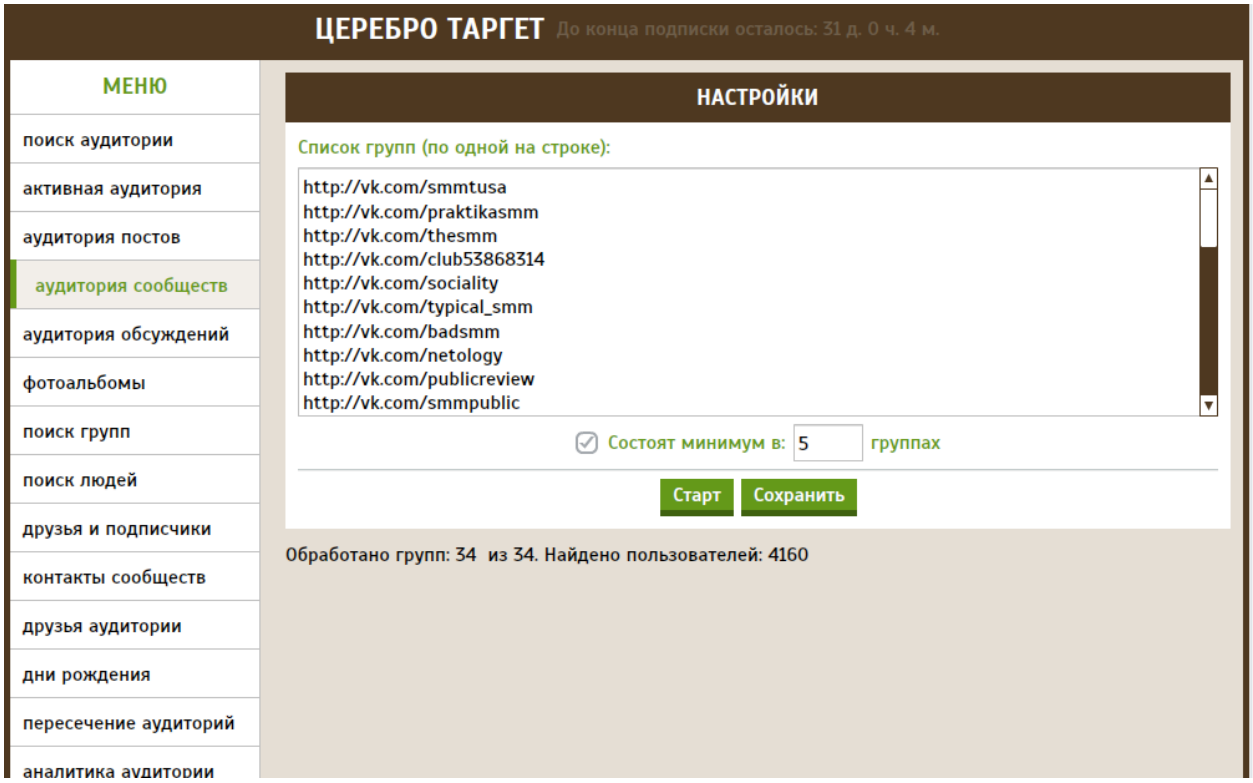

Рисунок 1 – интерфейс приложения «Церебро»

1.3.2 Анализ приложения «Сегменто-Таргет»

Второй по популярности аналог «Сегменто-Таргет» имеет ряд недостатков: за облачную версию приходится изрядно переплачивать, крайне сложный пользовательский интерфейс и некачественную работу самих парсеров.

В данном приложение, в отличии от приложения «Церебро» уже присутствует облачная версия, но за нее необходимо «доплатить». Сам интерфейс достаточно сложный и без внимательного прочтения инструкции разобраться достаточно трудно. Самый большой минус этого приложения – это парсеры, которые не фильтруют выходные данные. В результате в отчете может оказаться «мусор», который может помешать настройке таргетированной рекламы и его придется удалять вручную.

Из плюсов можно отметить: наличие тестового периода и достаточно высокую скорость работы. На рисунке 2 представлен интерфейс приложения «Сегменто-Таргет».

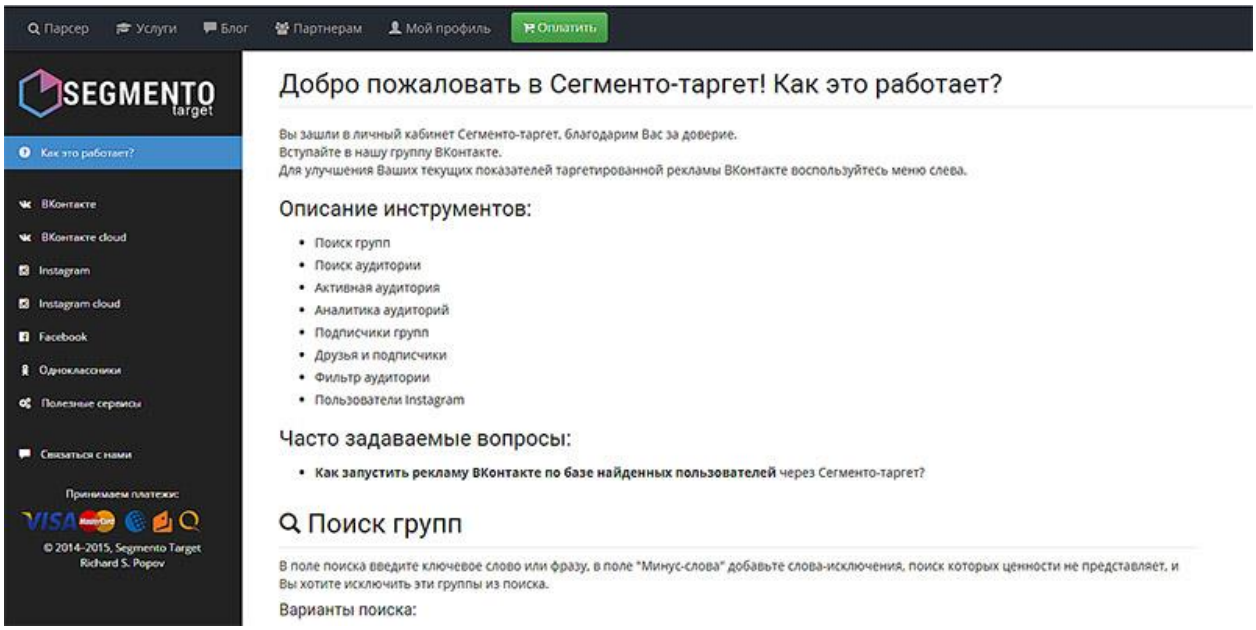

Рисунок 2 – интерфейс приложения «Сегменто-Таргет»

1.3.3 Анализ приложения «Retarget.sexy»

Третий аналог мало известен, но он имеет достаточно широкий функционал, хорошую скорость работы, удобный интерфейс и тестовый период. Минусы: отсутствие технической поддержки, базы обучающих материалов и сформированного сообщества клиентов сервиса. Так же предполагается, что после завершения тестирования всех функций, стоимость доступа к приложению значительно возрастет. На рисунке 3 представлен интерфейс приложения «Retarget.sexy».

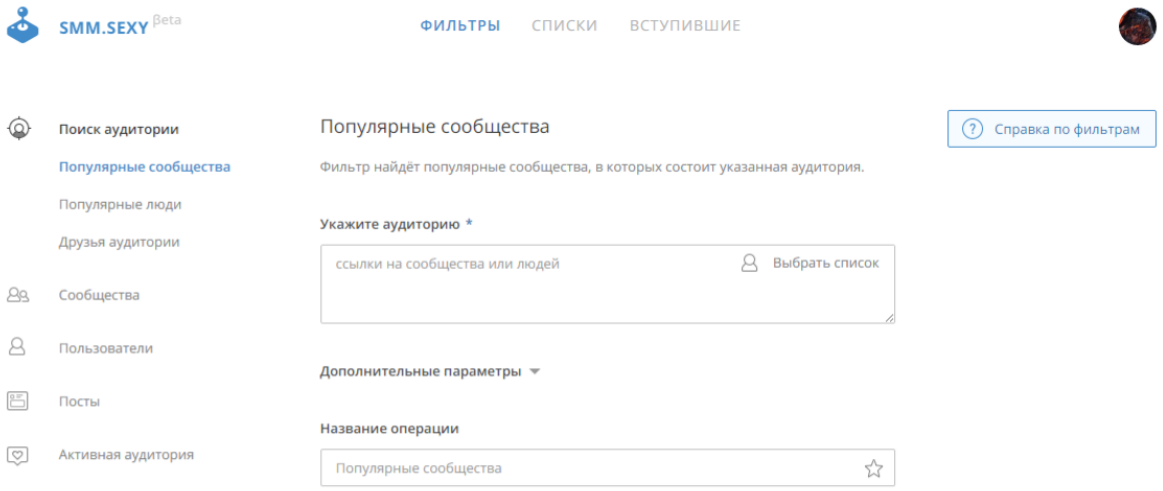

Рисунок 3 – интерфейс приложения «Retarget.sexy»

1.3.4 Выводы

Все рассмотренные приложения решают схожие задачи и предназначены для одной и той же целевой аудитории. Однако большая часть приложений имеет неудовлетворительную скорость работы и неудобный интерфейс. Приложения без облачной версии, которая бы выполняла все вычисления на стороне сервера уже «морально» устарели, и пользователи активно переходят на облачные приложения, которые, как правило быстрее, т.к не зависят от мощности клиента и скорости интернет соединения. Исходя из анализа были выделены основные требования к разрабатываемому приложению, которые задокументированы в следующем разделе «Техническое задание».

2 ТЕХНИЧЕСКОЕ ЗАДАНИЕ

2.1 Наименование и основание для создания

2.1.1 Полное наименование

Полное наименование: облачное веб-приложение для сбора баз ретаргетинга «VK.RETARGET»

2.1.2 Основание для создания

Разработка ведется на основании «Распоряжения об утверждении тем выпускных квалификационных работ» утвержденного заведующим кафедрой «Информатика». Тема выпускной квалификационной работы: «Разработка облачного веб-приложения для сбора баз ретаргетинга».

2.2. Назначение и цели создания системы

2.2.1 Назначение разрабатываемой системы

Приложение «VK.RETARGET» предназначено для упрощения поиска целевой аудитории в социальной сети «ВКонтакте». Предназначено для smm-специалистов, таргетологов, маркетологов, менеджеров по рекламе и собственников малого бизнеса.

2.2.2 Цели создания системы

Основной целью создания приложения «VK.RETARGET» является удовлетворение возросшего спроса к системам подобного вида с перспективой получения систематической прибыли.

2.3. Требования к системе

Разрабатываемая Система должна учитывать следующие основные принципы:

Открытость. Система должна использовать общедоступные и специфицированные решения, протоколы и интерфейсы, что будет обеспечивать интеграцию создаваемых решений, как между собой, так и с внешними информационными системами.

Модульность. Система должна быть построена с использованием модульной архитектуры, подразумевающей реализацию основных функций в качестве отдельных модулей, обеспечивающих возможность их независимой модификации. Сбой в работе одного из модулей не должен приводить к полному прекращению функционирования Системы в целом.

Управляемость и конфигурирование. Механизмы управления должны обеспечивать управление, создаваемой Системой, на всех уровнях ее архитектуры: на уровне инфраструктуры, на функциональном уровне, на уровне представления данных.

Системность. Все взаимосвязанные подсистемы, создаваемой Системы, должны использовать единую методологию и отвечать единым принципам взаимодействия, надежности и управления.

Единство графического представления. При проектировании и разработке пользовательских интерфейсов должны использоваться общие принципы графического представления информации и организации доступа пользователей к функциональным возможностям и сервисам создаваемой системы.

Архитектура приложения должна быть разделена по уровням абстракции. База данных должна быть вынесена за пределы всего приложения и размещена на отдельном сервере(серверах). Пользовательский интерфейс должен быть так же вынесен на отдельный, быстрый сервер для обеспечения максимальной скорости взаимодействия с пользователями.

Бизнес-логика приложения должна быть разделена на две части. Первая часть – это логическая часть, которая связывает пользовательский интерфейс с базой данной и обеспечивает общее функционирование приложения. Вторая часть – это вычислительная часть модуля «Задания», который отвечает за сбор и обработку больших объёмов данных полученных от API «ВКонтакте».

2.4. Требования к структуре и функционированию системы

2.4.1 Перечень модулей, их назначение и основные характеристики

В состав приложения «VK.RETARGET» должны входить следующие модули:

- модуль «Аутентификация»;
- модуль «Задания»;
- модуль «Отчеты»;
- модуль «Оплата»;

В модуль «Аутентификация» подразумевает возможность аутентификации по логину/паролю, а также по протоколу OAuth в социальной сети «ВКонтакте».

В модуль «Задания» входят парсеры: сообщества, пользователи, активности. Парсер «сообщества» подразумевает предоставление возможности поиска целевых сообществ по ключевым словам, с сохранением результата. Парсер «пользователи» подразумевает предоставление возможности сохранения списка подписчиков из сообществ, которые были найдены парсером «сообщества». Парсер «активности» подразумевает поиск активных подписчиков определенного сообщества с последующим сохранением результата.

В модуль «Отчеты» входит возможность сохранения и предоставление результатов работы модуля «Задания».

В модуль «Оплата» подразумевает оплату доступа к приложению. Возможна оплата по промо-коду и бесплатное использование.

2.5. Требования к модулям системы

2.5.1 Требования к модулю «Аутентификация»

В модуль аутентификация входит 3 основных функции:

- Регистрация
- Аутентификация
- Аутентификация «ВКонтакте».

Функция «регистрация» предназначена для регистрации новых пользователей в системе с последующим предоставлением соответствующих прав доступа. По умолчанию, все новые пользователи наделяются правами по умолчанию без возможности внесения изменений в данные других пользователей.

Функция «аутентификация» предназначена для проверки учетных данных пользователя и предоставления определенных прав доступа. В данном случае к данным каждого конкретного пользователя.

Функция «аутентификация «ВКонтакте»» предназначена для получения токена доступа к API «ВКонтакте». Осуществляется сразу после аутентификации с самом приложении путем перенаправления на специальный интерфейс «ВКонтакте».

2.5.2 Требования к модулю «Задания»

Модуль «Задания» работает с API «ВКонтакте», на основе прав доступа, полученных в модуле «Аутентификация». При реализации должно быть предусмотрено 3 основных парсера: сообщества, пользователи, активности.

Парсер «сообщества» должен предоставлять возможность находить сообщества в социальной сети по заранее введенным ключевым запросам.

Парсер «пользователи» должен предоставлять возможность получать список участников, найденных в парсере «сообщества».

Парсер «активности» должен предоставлять возможность получать список участников сообщества, которые проявили хотя бы 1 активность в нем.

2.5.3 Требования к модулю «Отчеты»

Модуль «Задания» должен предоставлять возможность сохранить результаты работы модуля «Задания» в текстовом формате для последующего использования в рекламном кабинете «ВКонтакте». В листинге 1 представлен пример сохраненного отчета, который содержит идентификаторы пользователе социальной сети «ВКонтакте».

```
12582917
25165830
209715210
243269644
184549390
33554447
192937962
155189265
58720275
264241172
```

Листинг 1 – Образец содержания отчета, парсер «пользователи»

Отчет о работе модуля «Задания» - парсер «сообщества» должен содержать список найденных сообществ по введенным ключевым словам. В листинге 2 приведен пример отчета по ключевому слову «красноярск»:

```
https://new.vk.com/detisibiri
https://new.vk.com/kraschp
```

```
https://new.vk.com/live_kras
https://new.vk.com/ilovekrsk
https://new.vk.com/public78933942
https://new.vk.com/krasnoyarskfree
https://new.vk.com/krsk_overhear
https://new.vk.com/krsk24online
```

Листинг 2 – Образец содержания отчета, парсер «сообщества»

2.5.4 Требования к модулю «Оплата»

Модуль «Оплата» должен предоставлять возможность оплатить доступ к приложению удобным для конечного пользователя способом на один из возможных периодов:

- 1 месяц (без скидки)
- 3 месяца (скидка 5%)
- 6 месяцев (скидка 10%)
- 12 месяцев (скидка 25%)

Оплата должна происходить через платежный агрегатор «Единая Касса» и должен предоставлять возможность оплатить следующими способами:

- Картой visa или mastercard
- Web Money
- Яндекс.Деньги
- QIWI

Данный перечень платежных систем позволит пользователям оплачивать так, как им удобно, тем самым повысив лояльность к приложению.

Также должна быть предоставлена возможность оплатить доступ к приложению по промо-коду.

2.6. Требование к архитектуре системы

Разрабатываемая Система должна учитывать следующие основные принципы:

- Модульность

Приложение должно быть построено с использованием модульной архитектуры, подразумевающей реализацию основных функций в качестве отдельных модулей, обеспечивающих возможность их независимой модификации. Сбой в работе одного из модулей не должен приводить к полному прекращению функционирования приложения в целом.

- Масштабируемость

Архитектура приложения должна позволять увеличивать производительность подсистем, объемы хранимой и обрабатываемой информации без длительной остановки работы и значительной модификации программного кода приложения.

- Управляемость и конфигурирование

Механизмы управления должны обеспечивать управление создаваемого приложения на всех уровнях ее архитектуры: на уровне инфраструктуры, на функциональном уровне, на уровне представления данных.

- Системность

Все взаимосвязанные подсистемы создаваемого приложения должны использовать единую методологию и отвечать единым принципам взаимодействия, надежности и управления.

2.7. Требования к интерфейсу системы

Интерфейс пользователя должен удовлетворять следующим требованиям:

- Интерфейс пользователя должен быть выполнен в стиле material design;
- Блоки информации должны выводится по «островной» системе;
- У каждого блока должна быть своя иконка, наглядно показывающая, какая информация будет находится в этом блоке;
- Уведомления для пользователя должны быть разделены по цвету. Красные - ошибки, желтые – предупреждения, синие – уведомления, зеленые – положительные уведомления.

2.8. Требования к надежности системы

Система должна сохранять работоспособность и обеспечивать восстановление своих функций при возникновении следующих внештатных ситуаций:

- при сбоях в системе электроснабжения аппаратной части, приводящих к перезагрузке ОС, восстановление программы должно происходить после перезапуска ОС и запуска веб–приложения в браузере;
- при ошибках в работе аппаратных средств (кроме носителей данных и программ) восстановление функции системы возлагается на ОС;
- при ошибках, связанных с программным обеспечением (ОС и драйверы устройств), восстановление работоспособности возлагается на ОС.

2.9. Требования к техническому обеспечению

Техническое обеспечение системы должно максимально и наиболее эффективным образом использовать существующие физические ресурсы компьютерных систем.

В состав комплекса должны следующие технические средства:

- Серверы БД;
- Веб сервер;
- ПК пользователей.

3 Описание и обоснование выбора инструмента для разработки программного обеспечения

3.1 Требования к инструментам серверной части приложения

В этой главе будут выдвигаться требования к технологической платформе приложения (программной составляющей), и в соответствующих подразделах будет произведен обзор существующих решений и выбор наиболее оптимального варианта.

3.1.1 Система управления базами данных

Система управления базой данных (СУБД) представляет собой программное обеспечение, которое управляет всем доступом к базе данных.

Для разрабатываемой системы, надежность и производительность СУБД является критическим моментом, поскольку приложение предполагает наличие большого количества записей в БД, а так же большое количество пользователей, чьи действия так или иначе будут связаны с манипулированием информации из БД. Основные требования к СУБД:

- управление данными во внешней памяти (на дисках);
- управление данными в оперативной памяти с использованием дискового кэша;
- журнализация изменений, резервное копирование и восстановление базы данных после сбоев;
- поддержка языков БД (язык определения данных, язык манипулирования данными);
- поддержка реляционной модели хранения данных;
- поддержка распределенного режима;
- поддержка клиент– серверной архитектуры взаимодействия;
- наличие внешних утилит для управления СУБД;
- низкая стоимость.

3.1.1.1 Обзор современных СУБД

В мире существует множество СУБД, обзор их всех не входит в рамки этой работы. Рассмотрены будут наиболее популярные СУБД, которые имеют многолетнюю историю разработки и поддержки.

3.1.1.1.1 Firebird

Firebird (FirebirdSQL) — компактная, кроссплатформенная, свободная система управления базами данных (СУБД), работающая на GNU/Linux, Microsoft Windows и разнообразных Unix платформах.

В качестве преимуществ Firebird можно отметить многоверсионную архитектуру, обеспечивающую параллельную обработку оперативных и аналитических запросов (это возможно потому, что читающие пользователи не блокируют пишущих), компактность (дистрибутив 5Mb), высокую эффективность и мощную языковую поддержку для хранимых процедур и триггеров.

Среди недостатков: отсутствие кеша, результатов запросов, полнотекстовых индексов.

Стоимость — бесплатно.

3.1.1.1.2 MySQL

MySQL — свободная система управления базами данных (СУБД). MySQL является собственностью компании Sun Microsystems, осуществляющей разработку и поддержку приложения. Распространяется под GNU General Public License и под собственной коммерческой лицензией, на выбор. Помимо этого разработчики создают функциональность по заказу лицензионных пользователей, именно благодаря такому заказу почти в самых ранних версиях появился механизм репликации.

Стоимость простой лицензии MySQL Enterprise на год — 600 у.е., или 18 000 рублей.

3.1.1.1.3 PostgreSQL

PostgreSQL — свободная объектно-реляционная система управления базами данных (СУБД). Существует в реализациях для множества UNIX-подобных платформ, включая AIX, различные BSD-системы, HP-UX, IRIX, Linux, Mac OS X, Solaris/OpenSolaris, Tru64, QNX, а также для Microsoft Windows. PostgreSQL ведёт свою «родословную» от некоммерческой СУБД Postgres, разработанной, как и многие open-source проекты, в Калифорнийском университете в Беркли. К разработке Postgres, начавшейся в 1986 году, имел непосредственное отношение Майкл Стоунбрейкер, руководитель более раннего проекта Ingres, на тот момент уже приобретённого компанией Computer Associates. Само название «Postgres» расшифровывалось как «Post Ingres», соответственно, при создании Postgres были применены многие уже ранее сделанные наработки.

Сильными сторонами PostgreSQL считаются:

- высокопроизводительные и надёжные механизмы транзакций и репликации;
- расширяемая система встроенных языков программирования: в стандартной поставке поддерживаются PL/pgSQL, PL/Perl, PL/Python и PL/Tcl; дополнительно можно использовать PL/Java, PL/PHP, PL/Py, PL/R, PL/Ruby, PL/Scheme, PL/sh и PL/V8, а также имеется поддержка загрузки C-совместимых модулей;
- наследование;
- легкая расширяемость.

Стоимость — бесплатно.

3.1.1.2 Выбор СУБД

PostgreSQL является самым профессиональным из всех трех рассмотренных СУБД. Она свободно распространяемая и максимально соответствует стандартам SQL. PostgreSQL или Postgres стараются полностью применять ANSI/ISO SQL стандарты своевременно с выходом новых версий.

От других СУБД PostgreSQL отличается поддержкой востребованного объектно-ориентированного и/или реляционного подхода к базам данных. Например, полная поддержка надежных транзакций, т.е. атомарность, последовательность, изоляционность, прочность (Atomicity, Consistency, Isolation, Durability (ACID).) Благодаря мощным технологиям Postgre очень производительна. Параллельность достигнута не за счет блокировки операций чтения, а благодаря реализации управления многовариантным параллелизмом (MVCC), что также обеспечивает соответствие ACID. PostgreSQL очень легко расширять своими процедурами, которые называются хранимые процедуры. Эти функции упрощают использование постоянно повторяемых операций.

Хотя PostgreSQL и не может похвастаться большой популярностью в отличии от MySQL, существует довольно большое число приложений облегчающих работу с PostgreSQL, несмотря на всю мощность функционала. Сейчас довольно легко установить эту СУБД, используя стандартные менеджеры пакетов операционных систем.

Кроме этого, использование PostgreSQL, является стандартом для приложений, написанных на фрейворке Ruby on Rails. Это означает наличие достаточного количества справочной литературы и развитое сообщество пользователей.

3.1.2 Языки программирования

Приложение состоит из двух равноценных частей: сервера и клиента. Использование одного и того же языка программирования чаще всего является или невозможным, или максимально нерациональным. В процессе разработки были выбраны наиболее оптимальные и удобные языки программирования для решения поставленных задач.

Серверная часть приложения написана на языке Ruby 2.1, в частности Ruby on Rails 4.2 – фреймворк, написанный на языке программирования Ruby, реализует архитектурный шаблон Model-View-Controller для веб-приложений, а также обеспечивает их интеграцию с веб-сервером и сервером баз данных. Является открытым программным обеспечением и распространяется под лицензией MIT.

Для «общения» с базой данных фреймворк Ruby on Rails предоставляет хороший интерфейс Active Record. Active Record выполнит запросы в базу данных за нас, он совместим с большинством СУБД (MySQL, PostgreSQL и SQLite - это только некоторые из них). Независимо от того, какая используется СУБД, формат методов Active Record будет всегда одинаковый. Это оптимальный выбор в условиях ограниченного времени выделенного на разработку приложения.

Для построения пользовательского интерфейса выбраны стандартные решения в виде связки HTML, CSS, JavaScript. За основу был взят фреймворк BootStrap – интуитивно простой и в тоже время мощный интерфейсный фрейморк, повышающий скорость, и облегчающий разработку web-приложений.

3.1.3 Веб– сервер

Веб-сервер — это сервер, принимающий HTTP– запросы от клиентов, обычно веб-браузеров, и выдающий им HTTP– ответы, обычно вместе с HTML – страницей, изображением, файлом, медиа– потоком или другими данными. Веб-серверы — основа Всемирной паутины.

Основным требованием для веб-сервера является обеспечение совместной работы всех предыдущих выбранных технологических решений.

3.1.3.1 Обзор современных веб-серверов

На данный момент на рынке существует огромное количество различных веб-серверов. В этом разделе мы рассмотрим наиболее популярные, бесплатные решения, подходящие для реализации данного приложения.

3.1.3.1.1 Обзор веб-сервера Ruby on Rails + Apache

Apache HTTP-сервер – свободный веб-сервер. По статистике, на май 2009 года используется в 46% всех веб-серверов, что делает его самым популярным веб--сервером. Версия 2.2.x обладает проверенной годами и миллионами пользователей стабильностью и надежностью. Сервер имеет большое количество модулей для работы со многими серверными технологиями.

3.1.3.1.2 Обзор веб-сервера Ruby on Rails + Nginx + Unicorn

Nginx — простой, быстрый и надёжный сервер, не перегруженный функциями. Применение nginx целесообразно прежде всего для статических веб-сайтов и как прокси-сервера перед динамическими сайтами. Основные возможности:

- обслуживание статических запросов, индексных файлов, автоматическое создание списка файлов, кеш дескрипторов открытых файлов;

- акселерированное проксирование без кэширования, простое распределение нагрузки и отказоустойчивость;
- поддержка кеширования при акселерированном проксировании и FastCGI;
- акселерированная поддержка FastCGI и memcached серверов, простое распределение нагрузки и отказоустойчивость;
- модульность, фильтры, в том числе сжатие (gzip), byte-ranges (докачка), chunked ответы, HTTP-аутентификация, SSI-фильтр;
- несколько подзапросов на одной странице, обрабатываемые в SSI-фильтре через прокси или FastCGI, выполняются параллельно;
- поддержка SSL.

Unicorn — свободный веб-сервер, написанный на языке Ruby и предназначенный для запуска Rack-приложений, в том числе, написанных на фреймворках Ruby on Rails или Sinatra. Unicorn может запускаться только под UNIX-подобными ОС (в том числе — Linux, FreeBSD, Mac OS), так как использует fork для создания воркеров. Поддерживает Ruby версии 1.8 и выше, но из-за более дружественного сборщика мусора рекомендуется использовать версии начиная с 2.0. Поддержка jRuby и других альтернативных реализаций Ruby не предусмотрена, в разработке — совместимость с Rubinius.

При запуске Unicorn создает множество клонов родительского процесса, каждый из которых используется в качестве однопоточного воркера. Каждый воркер в каждый момент времени обслуживает только одно соединение. После обработки и передачи данных клиенту воркер высвобождается и готов обслуживать следующего. Все воркеры подключаются к одному общему сокету (shared socket) или порту. Таким

образом, максимальное количество одновременных обрабатываемых запросов равно числу воркеров.

Сервер nginx конфигурируется как прокси-сервер и выступает в роли «менеджера» запросов, распределяя входящие запросы между запущенными копиями unicorn. Тем самым решается несколько проблем:

- отказоусточивость;
- масштабируемость;
- проблема «медленного клиента».

Отказоустойчивость достигается за счет того, что в случае возникновения проблем с 1 запросом (т.е 1 клиентом 1 копией приложения unicorn) остальные копии приложения продолжают работать, а по истечению некоторого времени «зависший» потом самостоятельно перезапускается и продолжает свою работу в нормальном режиме.

Масштабируемость заключается в возможности без особых проблем добавить «мощности» на основной сервер и запустить дополнительные копии приложения.

Проблема «медленного клиента» решается за счет использования прокси-сервера nginx – запрос не передается копии приложения на unicorn до тех пор, пока не будет получен полностью. Тем самым «медленные клиенты» не занимают лишних вычислительных мощностей, а переходят в режим ожидания до тех пор, пока входные данные не будут получены полностью.

3.1.3.1.3 Обзор веб–сервера Ruby on Rails + WEBrick

WEBrick — это библиотека, написанная на языке программирования Ruby и предоставляющая набор базовых HTTP-сервисов. WEBrick была написана Takahashi Masayoshi в соавторстве с Gotou Yuuzou, а также при участии других разработчиков со всего мира, в соответствии с моделью открытого программного обеспечения. Библиотека также содержит код для создания базовых сервисов, работающих и по другим протоколам, нежели по HTTP.

Библиотека используется программным каркасом Ruby on Rails для тестирования приложений в процессе их разработки. Несмотря на свою популярность, WEBrick заслуживает критики за полное отсутствие документации внутри своего кода. Несмотря на это, WEBrick поставляется как сервер по умолчанию во фреймворке Ruby on Rails и используется многими программистами как основной сервер, несмотря на отсутствие элементарной псевдо-многопоточности.

3.1.3.2 Выбор веб– сервера

В разрабатываемом приложении веб-сервер WEBrick используется по умолчанию для разработки и тестирования приложения, т.к поставляется «из коробки» и не требует дополнительной настройки.

В «production» среде в приложении используется связка Nginx + Unicorn, так как данное решение бесплатное, хорошо документировано и не требует сложной настройки, а так же решает множество актуальных проблем, среди которых реализация многопоточности, без необходимости написания соответствующего программного кода.

3.2 Требования к клиентской части приложения

Клиентская часть должна состоять из следующих компонент:

- Программное обеспечение: одна из операционных систем, Windows XP и выше, любой дистрибутив Linux с поддержкой графического режима (Ubuntu 7.04, Red Hat Linux, ALT Linux, PlayStation 3 OS и другие), MacOS X и выше;
- Любой интернет проводник (браузер) поддерживающий: HTML 1.0, JavaScript;
- Доступ к сети Интернет со скоростью не ниже 512 Кбит\сек.

4 Моделирование и проектирование программного обеспечения

4.1 Архитектура проектируемого приложения

В основу приложения заложен модульный принцип построения приложения в полной мере реализующий паттерн MVC(модель-представление-контроллер). При необходимости части приложения могут быть разделены и вынесены на отдельные сервера.

Сервер nginx выполняет роль прокси-сервера и статик-сервера, т.е выполняет роль «менеджера запросов», распределяя запросы на динамические данные между подпроцессами unicorn, и передавая готовые статические страницы клиенту, в случае, если таковые были запрошены. На рисунке 4 представлена архитектура разрабатываемого приложения.

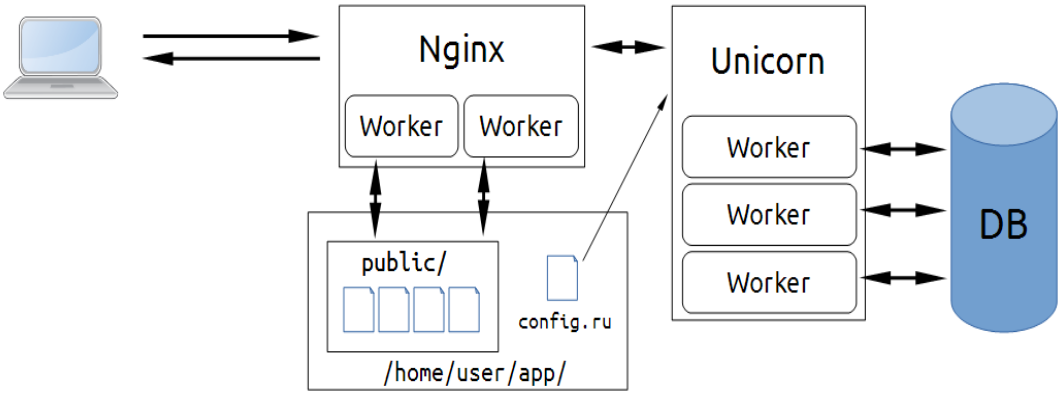

Рисунок 4 – схема обработки входящих запросов.

За управление приложением отвечают несколько основных контролеров построенных по REST принципу. Каждый такой контроллер подразумевает определенное количество стандартных экшенов, а именно: index, show, new, edit, create, update, delete.

Экшен index отвечает за стандартное представление произвольных данных в рамках ответственности данного контролера. Экшен show отвечает за представление определенного ресурса. Экшен new отвечает за предоставление интерфейса создания ресурса. Экшен edit отвечает за предоставление интерфейса для редактирования ресурса. Экшен create отвечает за создание ресурса и его запись в базу данных. Экшен update отвечает за обновление ресурса и внесение изменений в базу данных. Экшен delete отвечает за удаление ресурса.

Разработанная архитектура приложения позволяет легко масштабировать в производительности любой модуль, а так же делает их независимыми друг от друга, что существенно повышает надежность и стабильность приложения. Эти свойства, а также тот факт, что все вычисления производятся на стороне сервера, позволяют называть приложение «облачным».

На рисунке 5 представлена схема обработки входящих запросов.

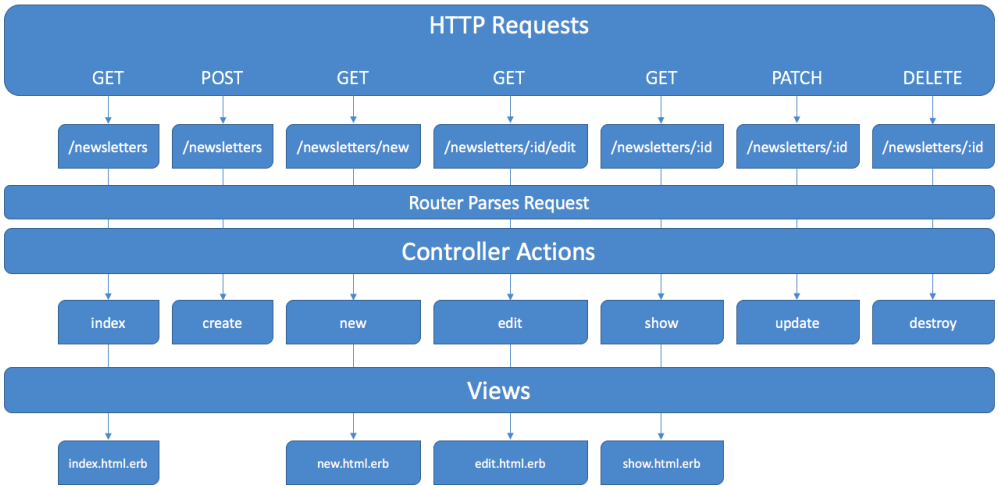

Рисунок 5 – Схема обработки запросов.

4.2 Построение диаграммы вариантов использования

Ниже описаны основные варианты использования системой: регистрация, аутентификация, добавление задания, просмотр отчетов и оплата.

Диаграмма вариантов использования для процесса аутентификации или регистрации пользователя.

Первый вариант использования описывает процесс регистрации или аутентификации в приложении. На главной странице пользователю предлагается или войти, используя свою пару логин:пароль или зарегистрироваться используя свой адрес электронной почты. Вне зависимости от того, регистрируется пользователь или входит по своим учетным данным, система делает автоматическое перенаправление пользователя на специальный интерфейс социальной сети, где пользователю предлагается подтвердить запрашиваемые права доступа, в случае если доступ будет подтвержден, пользователь будет автоматически перенаправлен в панель управления приложением. На рисунке 6 представлена диаграмма вариантов использования для модуля «Аутентификация».

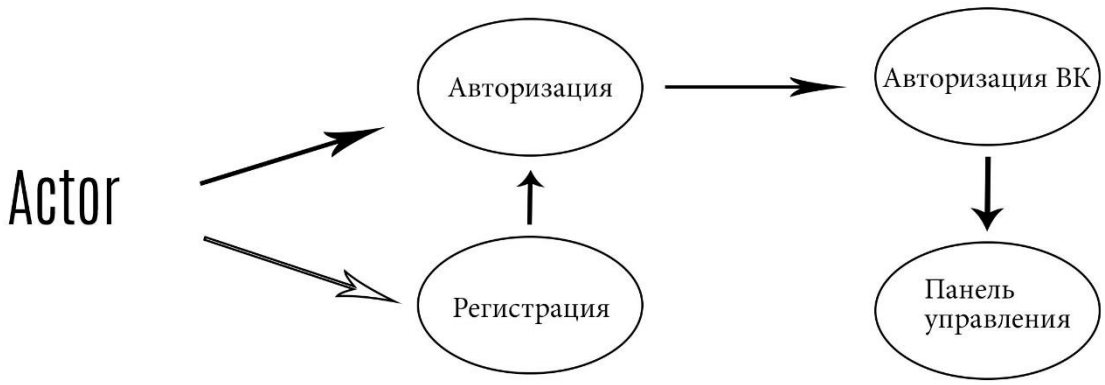

Рисунок 6 – Авторизация/регистрация

Второй вариант использования для модуля «Задания». Пользователь переходит в специальный раздел, где отображаются все текущие и завершенные задания. Ему предлагается добавить новое задание с помощью специальной формы, где можно выбрать нужные параметры, тип задания, загрузить список из файла или выбирается из доступных в приложении и предлагается добавить задание на выполнение. На рисунке 7 представлена диаграмма вариантов использования для модуля «Задания».

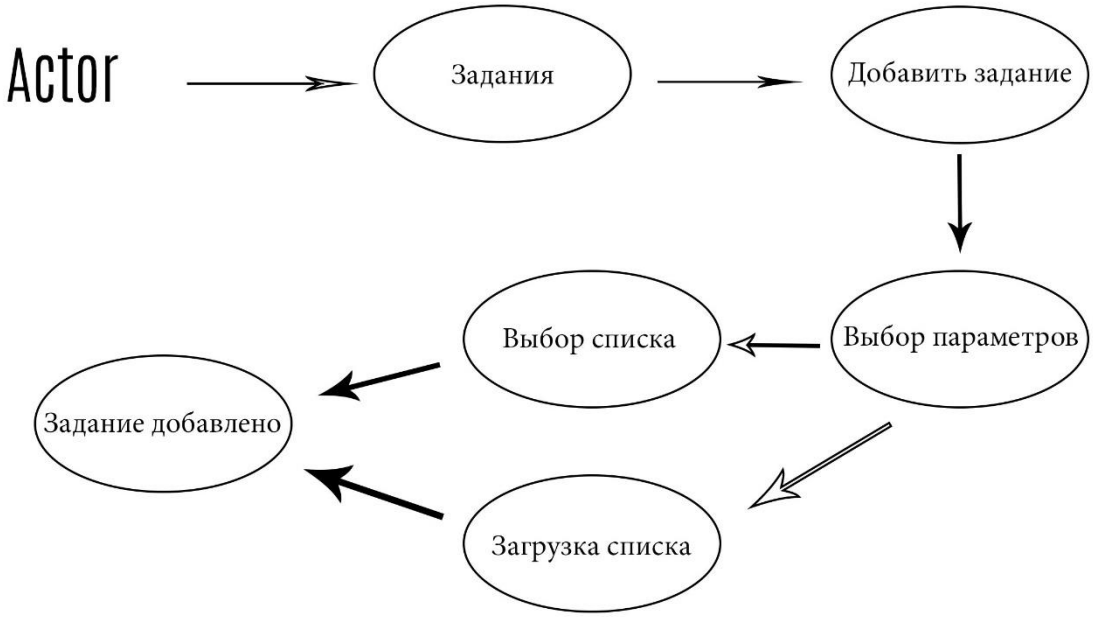

Рисунок 7 – Задания.

Вариант использования для модуля «Отчеты». В специальном разделе пользователю предлагается из списка завершенных заданий выбрать нужное и загрузить результаты работы парсера или на ПК пользователя или напрямую в рекламный кабинет «ВКонтакте». На рисунке 8 представлена диаграмма вариантов использования для модуля «Отчеты».

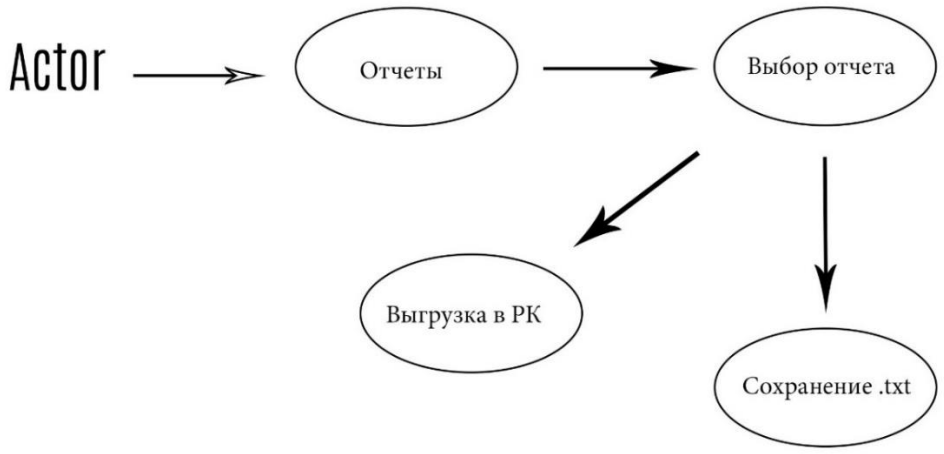

Рисунок 8 – работа с отчетами

Вариант использования для модуля «Оплата». После перехода в специальный раздел «Оплата» пользователю предлагается или ввести промо-код для активации тарифа или выбрать тариф и перейти к оплате доступа с помощью платежного агрегатора «Единая касса». На рисунке 8 представлена диаграмма вариантов использования для модуля «Отчеты».

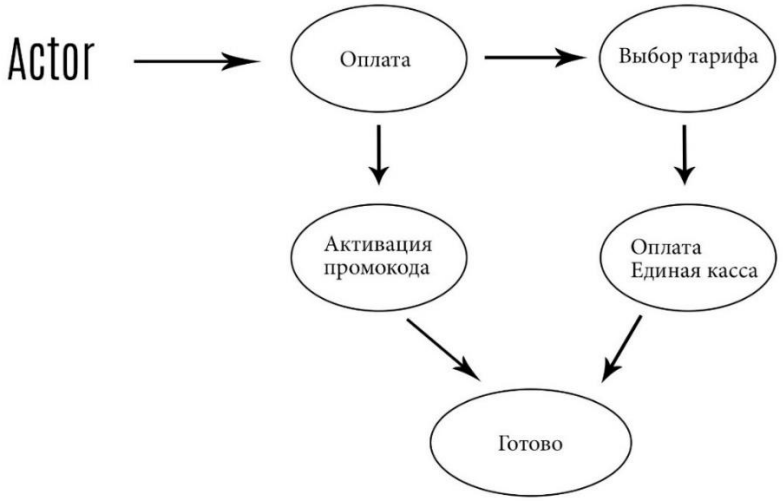

Рисунок 9 – Оплата

4.3 Описание информационной базы

База данных состоит из 6 основных таблиц: users, results, balances, delayed_jobs, tarif, promo.

Таблица users:

- email – почта пользователя
- encrypted_password – зашифрованный пароль
- reset_password_token – токен сброса пароля
- reset_password_sent_at – время запроса сброса пароля
- remember_created_at – время напоминания
- sign_in_count – количество входов в систему
- current_sign_in_at – дата текущего входа
- last_sign_in_at – дата последнего входа
- current_sign_in_ip – текущий ip
- last_sign_in_ip – последний ip
- created_at – дата создания записи
- updated_at – дата последнего обновления
- access_token – токен доступа к API ВК
- vk_id – идентификатор пользователя В

Таблица delayed_job служит электронной очередью для сервера заданий:

- priority – приоритет задания
- attempts – количество попыток выполнения
- handler – обработчик
- last_error – текст последней ошибки
- run_at – дата запуска (возможно в будущем)
- locked_at – дата блокировки от изменений
- failed_at – дата возникновения ошибки
- locked_by – кем заблокирован

- queue – имя очереди
- created_at – дата создания записи
- updated_at – дата последнего обновления

Таблица result служит для сохранения результатов работы парсеров. Таблица периодически полностью стирается. Схема таблицы:

- user_id – идентификатор пользователя, запустившего задачу
- task_id – идентификатор задачи
- task_name – имя задачи
- result_path – ссылка на результат

Таблица balances содержит информацию о тарифе и сроках доступа к сервису. По умолчанию сервис бесплатен. Схема таблицы:

- user_id – идентификатор пользователя
- tarif – тариф
- expires – дата окончания доступа

Таблица «promo» содержит список промо-кодов, информацию о его активности, о том, какой пользователь его активировал и какой тариф этот промо-код активирует. Схема таблицы:

- balance_id
- activated
- tarif

Таблица «tarif» содержит информацию о тарифе: его описание, стоимость, сроках действия. Схема таблицы:

- name
- description
- cost
- validity

На рисунке 10 представлена схема базы данных с расставленными связями.

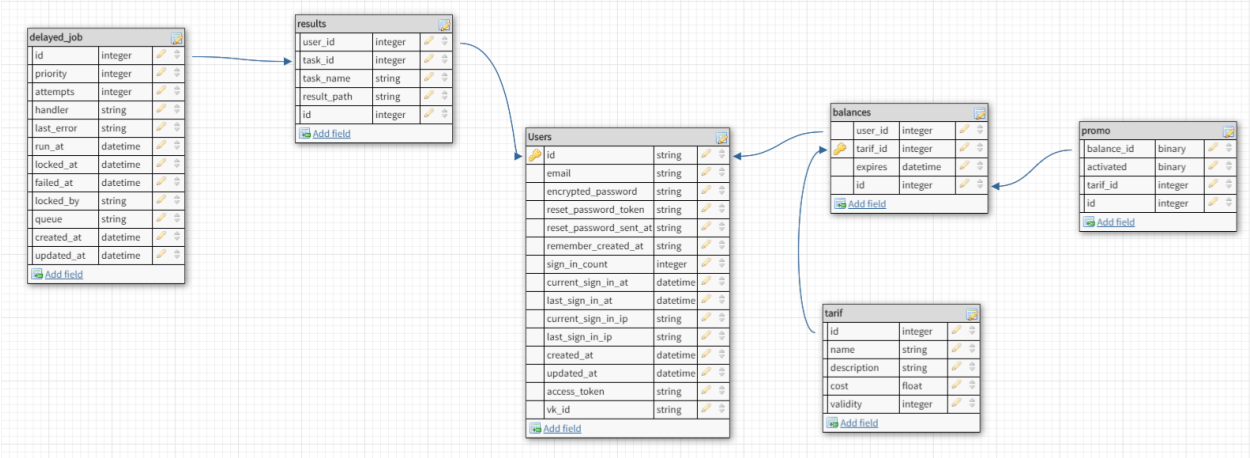

Рисунок 10 – Схема базы данных

4.5. Вывод

В данном разделе была проведена работа по проектированию разрабатываемой системы. Спроектирована и описана архитектура приложения, проведено моделирование базы данных, описана функциональность и поведение системы с помощью диаграмм вариантов использования.

5 Описание разработанного программного обеспечения

5.1 Модуль «Аутентификация»

Для доступа к приложению необходимо открыть интернет браузер и набрать адрес приложения. На локальном сервере это http://localhost:3000/.

Для входа в приложения необходимо ввести свой логин и пароль или пройти процедуру регистрации, нажав на соответствующую ссылку в правом верхнем углу формы.

На рисунке 11 представлена форма аутентификации.

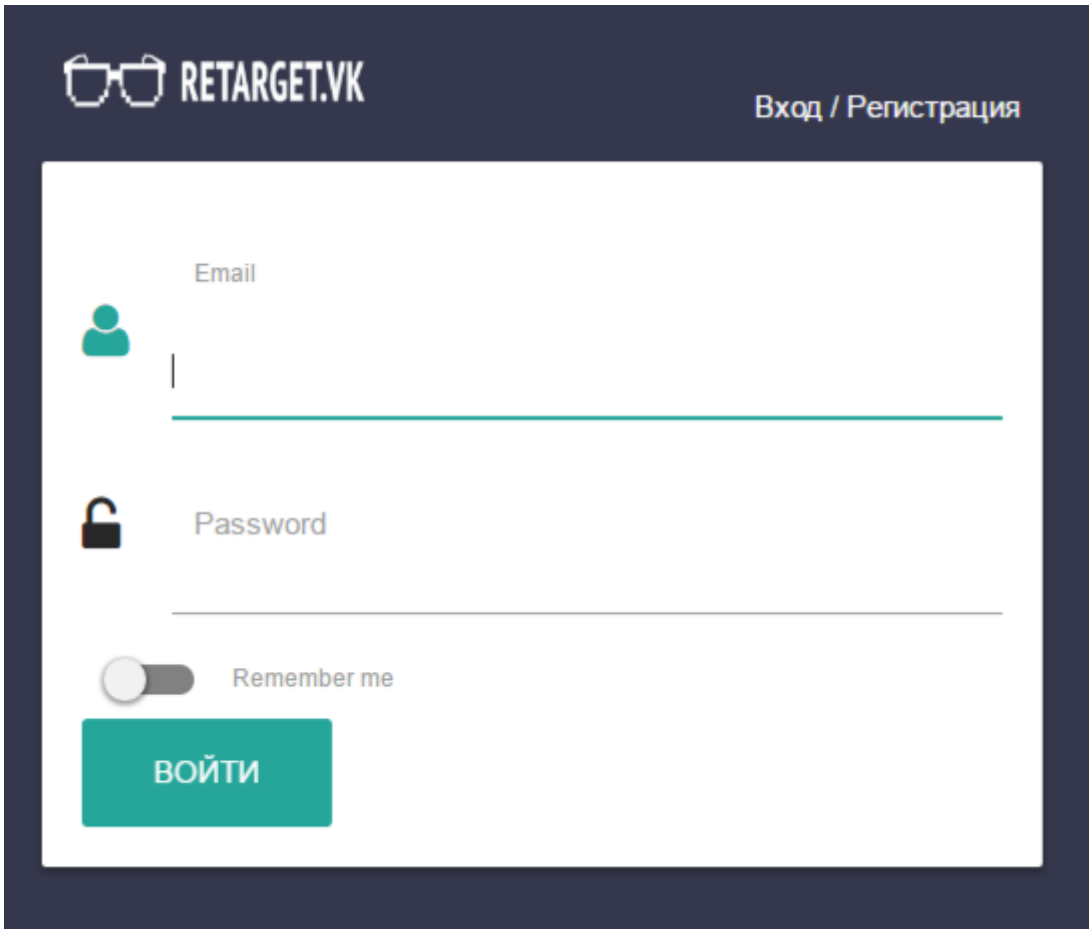

Рисунок 11 – форма авторизации с приложении.

Если, у пользователя нет учетных данных для входа в приложение, ему предлагается пройти несложную регистрацию в приложении. Для этого необходимо ввести:

- email,
- пароль,
- подтверждение пароля.

Если все данные были введены верно, пользователь будет перенаправлен в панель управления приложением. Если данные введены не верно, например, введенные пароль не совпадают, пользователю будет показано соответствующее уведомление с описанием ошибки. На рисунке 12 представлена форма регистрации в приложении.

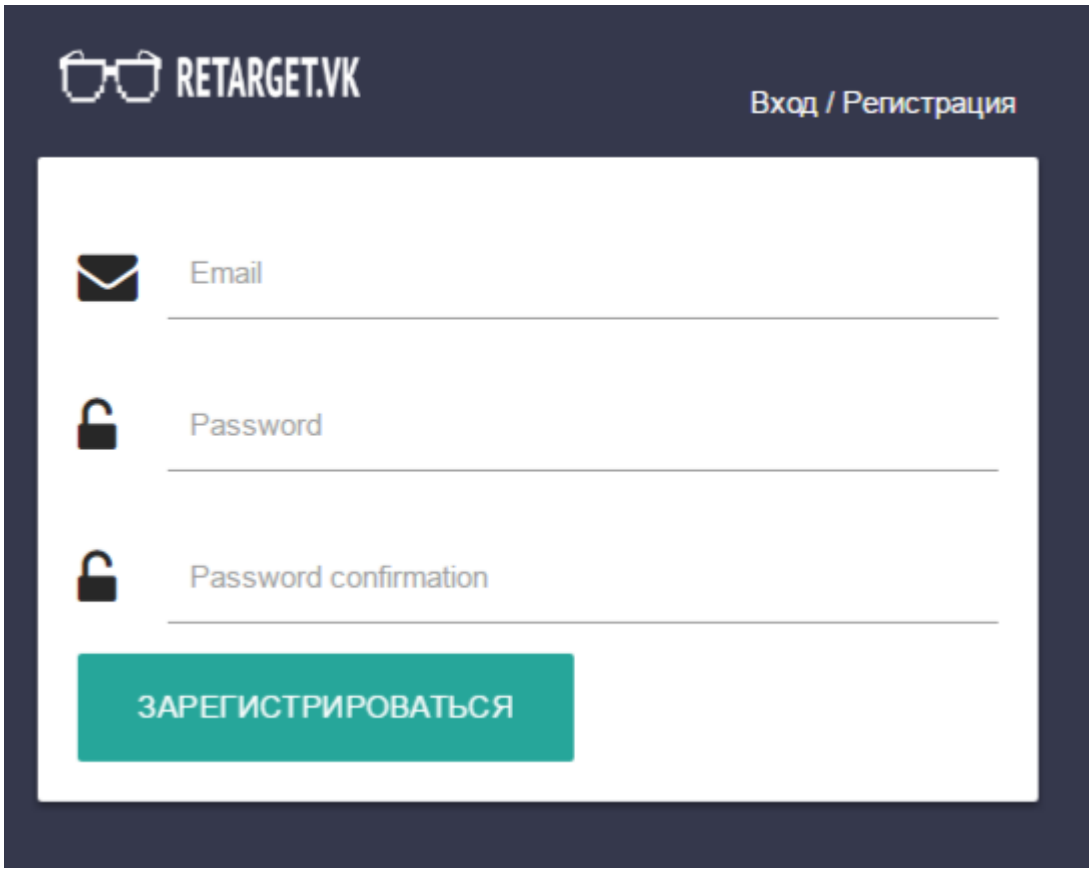

Рисунок 12 – форма регистрации в приложении.

5.2 Панель управления

Панель управления приложением позволяет получить основную информацию о текущих и выполненных заданиях, загрузить отчеты, просмотреть статистику и профиль пользователя. Интерфейс выполнен в виде «островов», что упрощает восприятие информации. На рисунке 13 представлен интерфейс панели управления приложением.

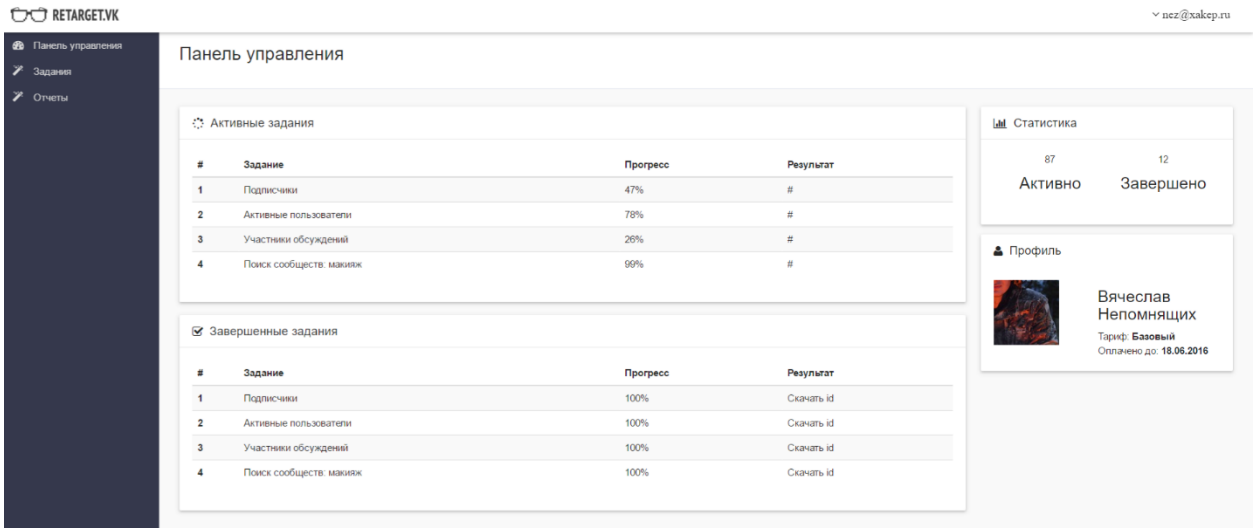

Рисунок 13 – интерфейс панели управления приложением.

5.3 Модуль «Задания»

Основа модуля «Задания» - это специальная форма, которая позволяет ввести или выбрать список исходных данных для обработки парсером и выбрать необходимый парсер. Кнопка «Добавить задание» добавляет задание в очередь на выполнение. На рисунке 14 представлена форма добавления задания в систему.

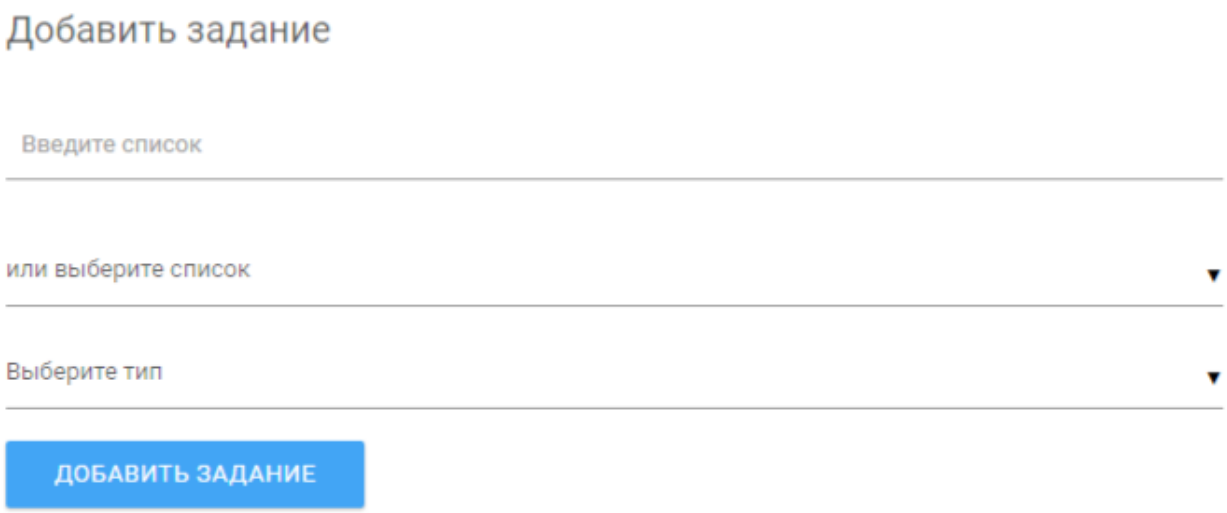

Рисунок 14 – форма добавления задания.

5.4 Модуль «Отчеты»

Модуль отчеты позволяет просмотреть все завершенные задания, и при необходимости скачать результат работы на персональный компьютер. На рисунке 15 представлен интерфейс модуля «Отчеты».

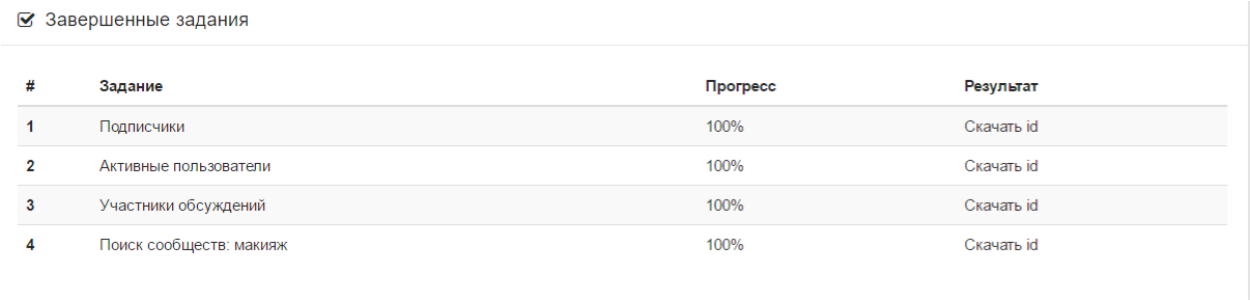

Рисунок 15 – интерфейс модуля «Отчеты».

5.5 Модуль «Оплата»

Модуль «Оплата» не имеет своего интерфейса. Переход к оплате происходит непосредственно из панели управления путем нажатия на специальную кнопку, т.к в данный момент в приложении активен только 1 вариант оплаты – сроком на 1 месяц. На рисунке 16 представлен интерфейс платежного агрегатора «Единая касса».

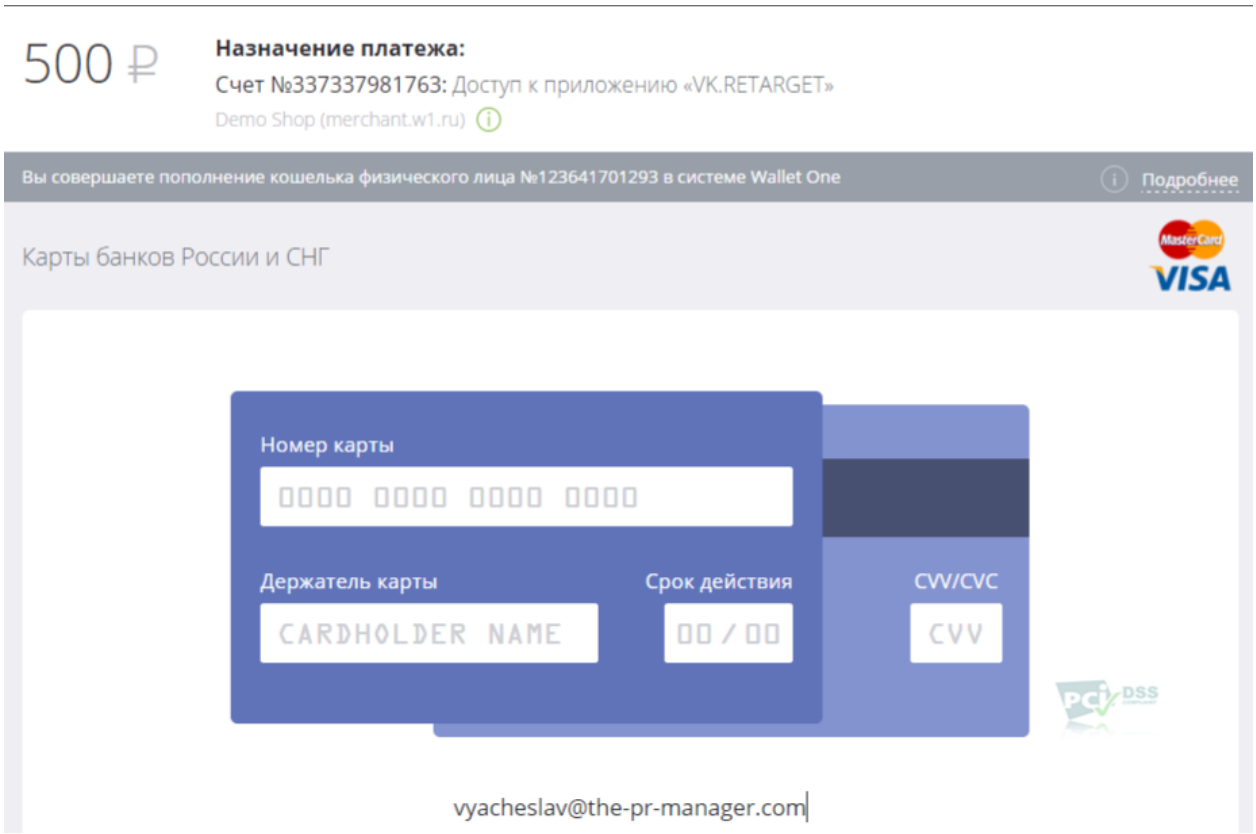

Рисунок 16 – форма оплаты доступа к приложению.

ЗАКЛЮЧЕНИЕ

Целью данной работы являлась разработка приложения для сбора баз ретаргентинга.

В работе был проведен анализ предметной области работы приложений подобного типа. Осуществлен анализ существующих решений на рынке ПО для сбора баз ретаргетинга. Были сформированы требования к системам подобного типа.

Так же в рамках были решены следующие задачи:

- Спроектирована архитектура приложения
- Спроектирован пользовательский интерфейс, отвечающий современным требованиям пользователей
- Спроектировано и реализовано хранилище данных
- Выполнена разработка и тестирование приложения для сбора баз ретаргетинга и присвоено кодовое название «VK.RETARGET»

В ходе выполнения работы получены следующие навыки:

- Опыт работы с системами контроля версий(git)
- Опыт проектировки современных интерфейсов
- Опыт разработки модульных систем

Разработанное приложение планируется опубликовать для открытого тестирования в августе 2016 года. К концу 2016 года планируется запуск первых платных тарифов и ограничение бесплатного использования приложением.

СПИСОК СОКРАЩЕНИЙ

ПК – персональный компьютер

Экшен – метод контроллера

Контроллер – специальный класс, отвечающий за адресацию HTTP-запросов

ФИО – фамилия имя отчество

БД – база данных

ВК, «ВКонтакте» – российская социальная сеть

API ВК – специальный унифицированный интерфейс доступа к данным социальной сети.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Скринкасты, вопросы, тесты Ruby. Ссылка на электронный источник: http://ruby.hasbrains.org/
2. Скринкасты, вопросы, тесты Rails. Ссылка на электронный источник: http://rails.hasbrains.org/
3. Flexible authentication solution for Rails with Warden Ссылка на электронный источник: https://github.com/plataformatec/devise
4. Database based asynchronous priority queue system. Ссылка на электронный источник: https://github.com/collectiveidea/delayed_job
5. Ruby on Rails API. Ссылка на электронный источник: http://api.rubyonrails.org/
6. Ruby on Rails Tutorial: Изучение Rails на Примерах. Ссылка на электронный источник: http://railstutorial.ru/chapters/4_0/beginning
7. Связка rvm + Rails + Nginx + Unicorn или деплоим рельсы правильно. Ссылка на электронный источник: https://habrahabr.ru/post/120368/
8. Unicorn. Ссылка на электронный источник: https://github.com/defunkt/unicorn
9. Nginx. Ссылка на электронный источник: https://nginx.org/ru/
10. Bootstrap. Ссылка на электронный источник: http://getbootstrap.com/