Оглавление

[Реферат 4](#_Toc517874798)

[Введение 5](#_Toc517874799)

[1. Обзор предметной области поиска абитуриентов на базе технологий NLP и машинного обучения 5](#_Toc517874800)

[1.1 Описание предметной области поиска абитуриентов на базе технологий NLP и машинного обучения 5](#_Toc517874801)

[1.2 Выбор социальной сети для анализа 7](#_Toc517874802)

[1.3 Обзор аналогов сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 8](#_Toc517874803)

[1.4 Формулировка требований к сервису поиска абитуриентов на базе технологий NLP и машинного обучения 10](#_Toc517874804)

[2. Проектирование сервиса поиск абитуриентов на базе технологий NLP и машинного обучения 11](#_Toc517874805)

[2.1 Проектирование алгоритмов сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 11](#_Toc517874806)

[2.1 14](#_Toc517874807)

[2.2 Проектирование базы данных сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 21](#_Toc517874808)

[2.3 Проектирование физического представления сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 23](#_Toc517874809)

[3. Разработка сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 25](#_Toc517874810)

[3.1 Выбор языка программирования сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 25](#_Toc517874811)

[3.2 Выбор СУБД для сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 26](#_Toc517874812)

[3.3 Выбор модели для обучения сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 27](#_Toc517874813)

[3.4 Разработка интерфейса сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 28](#_Toc517874814)

[3.5 Разработка модели обучения сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 29](#_Toc517874815)

[3.6 Разработка сборщика матрицы для обучения сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 30](#_Toc517874816)

[3.7 Разработка сборщика пользовательских данных сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 30](#_Toc517874817)

[3.8 Разработка очистки музыкальных исполнителей сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 30](#_Toc517874818)

[3.9 Разработка разбиения на категории факультетов сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 30](#_Toc517874819)

[4. Тестирование сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 30](#_Toc517874820)

[4.1 Тестирование интерфейса сервиса поиска абитуриентов на базе технологий NLP и машинного обучения фокус-группой 30](#_Toc517874821)

[Заключение 33](#_Toc517874822)

[Список использованных источников 33](#_Toc517874823)

[Приложения 33](#_Toc517874824)

[Приложение А Техническое задание на создание сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 33](#_Toc517874825)

[Приложение Б Диаграмма вариантов использования 33](#_Toc517874826)

[Приложение В Диаграмма классов 33](#_Toc517874827)

[Приложение Г Диаграмма развёртывания 33](#_Toc517874828)

[Приложение Д Диаграмма базы данных сервиса поиска абитуриентов на базе технологий NLP и машинного обучения 33](#_Toc517874829)

# Реферат

Выпускная квалификационная работа, ##с., # рис., # табл., # источников, #прилож.

**Ключевые слова:** Python, NLP, база данных, #.

**Объектом** сервис для поиска абитуриентов.

**Цель работы** – разработка сервиса для поиска абитуриентов на базе технологий NLP и машинного обучения.

В процессе работы производилось определение требований к сервису и его реализация.

**Достигнутые технико-эксплуатационные показатели:** #

**Степень внедрения:** #

**Результат:** в результате разработки был получен полноценный работающий сервис поиска абитуриентов #.

# Введение

В 2008 году ученые из университета Хэриот-Уотт (Эдинбург) во главе с профессором Адрианом Нортом, главой кафедры прикладной психологии, решили проверить, связаны ли музыкальные предпочтения с интеллектом и характером слушателей [#]. Их исследования выявили очень интересную особенность: самые высокие результаты IQ-тестов - у любителей тяжелой музыки и рока. На основе данных о том, что музыка взаимосвязана с уровнем интеллекта, вывели гипотезу, что она так же взаимосвязана с предрасположенностью к какому-либо направлению обучения. Для подтверждения данной гипотезы было разработано программное обеспечение, которое на основе выборки студентов по разным направлениям, находит зависимости между ними.

# Обзор предметной области поиска абитуриентов на базе технологий NLP и машинного обучения

## Описание предметной области поиска абитуриентов на базе технологий NLP и машинного обучения

Определения:

* **Natural language processing (NLP)** - общее направление искусственного интеллекта и математической лингвистики. Оно изучает проблемы компьютерного анализа и синтеза естественных языков. Применительно к искусственному интеллекту анализ означает понимание языка, а синтез — генерацию грамотного текста. [#]
* Таргетинг (англ. target — цель) — рекламный механизм, позволяющий выделить из всей имеющейся аудитории только ту часть, которая удовлетворяет заданным критериям (целевую аудиторию), и показать рекламу именно ей.
* Датасет - идентифицированная совокупность физических записей, организованная одним из установленных в системе обработки данных способов и представляющая файлы или части файлов в среде хранения [5].

Между университетами (далее ВУЗ) происходит конкуренция за лучшие умы поступающих. На сегодняшний день финансирование вузов зависит от результатов приемной компании. Поэтому для ВУЗа важен каждый студент, но если этот студент учится на направлении, к которому у него нет предрасположенности, то вероятность его отчисления из ВУЗа велика. Поэтому для создания списка первоочередных абитуриентов нужен их анализ, который мы будем брать из социальной сети Вконтакте, на основе музыки, которую он слушает.

Стратегия проста, выявить школьников с предрасположенностью к определенному направлению обучаемом в ВУЗе и пригласить его на обучение.

Выявлен один прямой аналог Томского государственного университета, но данный продукт строит целевую модель поступающего не только по музыке, но и по «следам», оставленным в социальной сети Вконтакте, подробности не разглашаются. Косвенные аналоги – программы для таргетинга.

Постановка задач:

* Выявить зависимость между музыкой исследуемого и его предрасположенностью к определенному виду деятельности.
* Спроектировать структуры в БД
* Разработать структуру БД
* Подготовить датасет
* Спроектировать пользовательский интерфейс
* Разработать пользовательский интерфейс
* Спроектировать алгоритм системы
* Разработать скрипты для работы системы
* Проанализировать студентов ХМАО и разбить их на целевые группы по направлениям обучения
* Научить Систему распознавать абитуриента по профилю социальной сети ВКонтакте

Актуальность:

Тема актуальна, так как каждый ВУЗ (государственные и коммерческие) ведет борьбу за студентов. Такой подход позволит университету выходить за границы своего расположения и активно искать, и привлекать талантливые умы на территориях соседних регионов без существенных финансовых издержек.

«Худший формат дня открытых дверей, который можно придумать, — посадить в фойе тетушек пенсионного возраста с буклетами вуза. Это будет первый и последний день, когда абитуриент переступит порог этого института», — рассказывает психолог, карьерный консультант, программный директор Школы Осознанного Развития Елена Рагозина. — «Современные дети не понимают ту информацию, которую им доносят на днях открытых дверей, проходящих по классической схеме. Поэтому побеждают ВУЗы, распространяющие информацию о себе заблаговременно и не в традиционных направлениях».

## Выбор социальной сети для анализа

По данным ВЦИОМ (всероссийский центр изучения общественного мнения) самой популярной социальной сетью в России является «ВКонтакте» (далее Вконтакте). Так 42% российских интернет-пользователей практически ежедневно пользуются «Вконтакте» (в молодежных группах – 78% среди 18-24-летних, 54% среди 25-34-летних), 27% - «Одноклассниками» (среди 60-летних и старше – 40%). [https://wciom.ru/index.php?id=236&uid=116254] Ежемесячная аудитория «ВКонтакте» составляет более 97 000 000 человек. На основе данных исследований будем использовать социальную сеть Вконтакте

## Обзор аналогов сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

Определения:

* **Парсер** (от англ. *parser*) или **граббер** (от англ. *grabber*) — программное обеспечение, предназначенное для анализа и разбора исходных данных, с целью их обработки и дальнейшей использования в требуемом виде.
* **Парсинг –** автоматический анализ и сбор данных по заданному признаку.
* **Целевая группа, целевая аудитория** — термин, используемый в маркетинге или рекламе для обозначения группы людей, объединенных общими признаками, или объединенной ради какой-либо цели или задачи.

Критерии для сравнения аналогов:

* Цена
* Работа с API VK
* Тип ПО
* Скорость обработки
* Машинное обучение (Анализ данных)

В ходе анализа не было выявлено ни одного прямого аналога. В результате чего, был произведен поиск косвенных аналогов, занимающихся схожим функционалом.

Аналоги:

* Проект: «Поиск и привлечение абитуриентов с высоким потенциалом через социальную сеть» Томского государственного университета - далее Проект
* Сервис поиска целевой аудитории ВКонтакте vk.barkov.net
* Vk Audio Parser v2.2
* Яндекс.Музыка
* Pepper.ninja

В ходе сравнения аналогов, была составлена сравнительная таблица:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Критерии/ПО | Цена (руб/мес.) | Работа с музыкой | Тип ПО | Скорость обработки (пользователей/мин.) | Анализ данных |
| Проект | ➖ | ➖ | Неизвестен | Неизвестна | ✔ |
| vk.barkov.net | 390 | ✔,➖ (Только со страницы профиля) | Web-сервис | 3000 | ➖ |
| Vk Audio Parser v2.2 | 1200 | ✔ | Настольное ПО | 150 | ➖ |
| Pepper.ninja | 690 | ➖ | Web-сервис | 2000 | ➖ |
| Яндекс.Музыка | 100 | ✔ | Web-сервис | Персонально | ✔ |

Данные таблицы актуальный на 15 июня 2018 г.

Как видно из таблицы сравнения, одним из главных критериев был анализ данных, данному критерию удовлетворяют Яндекс.Музыка и Проект.

Из описания Проекта, реализуется схожая функция – поиск абитуриентов по мотивации к изучению предметов определенного направления (гуманитарные, точные и естественные науки).

Яндекс.Музыка на базе машинного обучения анализирует список музыки пользователя, его настроение и строит ежедневный трек лист. Так же Яндекс проводил исследование «У каждого поколения — своя музыка. Или нет», в котором анализировалась музыка для каждого поколения. В результате была выявлена зависимость, подробная информация в источниках литературы [#].

Настольное ПО Vk Audio Parser v2.2 работал с музыкой через API VK, на данный момент функционал API VK AUDIO закрыт, программа перестала работать. Интерфейс программы оставляет желать лучшего. Для работоспособности программы не нужен сервер.

vk.barkov.net и Pepper.ninja имеют приятный интерфейс и быструю скорость обработки. Скорость можно поднять, за счет другого тарифного плана.

Проанализировав аналоги было принято решение писать web-сервис. Разработка данного сервиса актуальна для университетов, прямых аналогов не имеет.

## Формулировка требований к сервису поиска абитуриентов на базе технологий NLP и машинного обучения

Разрабатываемая Система должна соответствовать следующим требованиям:

Функциональные требования:

* Составлять прогноз абитуриентов???
* Самообучение системы
* Работа с БД
* Обработка называний музыкальных групп
* Работа с файлом JSON

Нефункциональные требования:

* Системные требования:
  + ОЗУ – 6 Гб
  + Частота процессора – 2 ГГц
* Требование к персоналу:
  + Уровень пользования ПК – уверенный
* Требования к пользовательскому интерфейсу:
  + Минимальный интерфейс
  + Возможность выбора файла

# Проектирование сервиса поиск абитуриентов на базе технологий NLP и машинного обучения

## Проектирование алгоритмов сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

Разработка начинается с выбора модели жизненного цикла программного обеспечения. Воспользовавшись рекомендациями Института качества программного обеспечения SQI были заполнены таблицы, на основе которых был сделан вывод о необходимой модели.

Таблица 3.1

Выбор модели жизненного цикла на основе характеристик требований

|  |  |  |
| --- | --- | --- |
| Номер критерия | Критерии категории требований | Ответ на критерий |
| 1. | Являются ли требования к проекту легко определимыми и реализуемыми? | Нет |
| 2. | Могут ли требования быть сформулированы в начале ЖЦ? | Да |
| 3. | Часто ли будут изменяться требования на протяжении ЖЦ? | Нет |
| 4. | Нужно ли демонстрировать требования с целью их определения? | Да |
| 5. | Требуется ли проверка концепции программного средства или системы? | Нет |
| 6. | Будут ли требования изменяться или уточняться с ростом сложности системы (программного средства) в ЖЦ? | Да |
| 7. | Нужно ли реализовать основные требования на ранних этапах разработки? | Да |

Таблица 3.2

Выбор модели жизненного цикла на основе характеристик команды разработчиков

|  |  |  |
| --- | --- | --- |
| Номер критерия | Критерии категории команды разработчиков проекта | Ответ на критерий |
| 1. | Являются ли проблемы предметной области проекта новыми для большинства разработчиков? | Да |
| 2. | Являются ли инструментальные средства, используемые в проекте, новыми для большинства разработчиков? | Да |
| 3. | Изменяются ли роли участников проекта на протяжении ЖЦ? | Нет |
| 4. | Является ли структура процесса разработки более значимой для разработчиков, чем гибкость? | Нет |
| 5. | Важна ли легкость распределения человеческих ресурсов проекта? | Да |
| 6. | Приемлет ли команда разработчиков оценки, проверки, стадии разработки? | Да |

Таблица 3.3

Выбор модели жизненного цикла на основе характеристик коллектива пользователей

|  |  |  |
| --- | --- | --- |
| Номер критерия | Критерии категории коллектива пользователей | Ответ на критерий |
| 1. | Будет ли присутствие пользователей ограничено в ЖЦ разработки? | Да |
| 2. | Будут ли пользователи оценивать текущее состояние программного продукта (системы) в процессе разработки? | Да |
| 3. | Будут ли пользователи вовлечены во все фазы ЖЦ разработки? | Нет |
| 4. | Будет ли заказчик отслеживать ход выполнения проекта? | Да |

## 

Таблица 3.4

Выбор модели жизненного цикла на основе характеристик типа проектов и рисков

|  |  |  |
| --- | --- | --- |
| Номер критерия | Критерии категории коллектива пользователей | Ответ на критерий |
| 1. | Разрабатывается ли в проекте продукт нового для организации направления? | Да |
| 2. | Будет ли проект являться расширением существующей системы? | Да |
| 3. | Будет ли проект крупно- или среднемасштабным? | Да |
| 4. | Ожидается ли длительная эксплуатация продукта? | Да |
| 5. | Необходим ли высокий уровень надежности продукта проекта? | Да |
| 6. | Предполагается ли эволюция продукта проекта в течение ЖЦ? | Да |
| 7. | Велика ли вероятность изменения системы (продукта) на этапе сопровождения? | Да |
| 8. | Является ли график сжатым? | Да |
| 9. | Предполагается ли повторное использование компонентов? | Да |
| 10. | Являются ли достаточными ресурсы (время, деньги, инструменты, персонал)? | Нет |

На основании результатов, представленных в таблицах 3.1 – 3.4 для спроектированного ПО преимущественно подходит инкрементная модель жизненного цикла.

В инкрементной модели полные требования делятся на различные версии ПО. Каждая версия - это определенный этап сборки общего ПО. Каждый модуль проходит через фазы определения требований, проектирования, кодирования, внедрения и тестирования. Процесс циклов будет продолжаться до тех пор, пока не будут добавлены все необходимые функции и ПО не будет полностью готово. Схема инкрементной модели представлена на рисунке #.



Разработка программного обеспечения подразумевает под собой 3 фазы со своими подзадачами:

* Работа с базой данных
  + Проектирование БД
  + Первоначальный сбор данных
  + Очистка данных
* Построение модели ПО
  + Проектирование моделей ПО
  + Доработка сборщика данных
  + Доработка очистки данных
  + Машинное обучение
* Готовый сервис
  + Сбор всех модулей в один сервис

В ходе написания ВКР был спроектирован алгоритм подготовки датасета. Алгоритм подготовки датасета представлен на рисунке 1.



Рисунок # - алгоритм подготовки датасета

Первым шагом происходит сбор данных аудио исполнителей у пользователей социальной сети Вконтакте. Затем происходит визуальный обзор собранных данных, для выявления стоп-слов и мусора. После составления списка стоп-слов происходит предварительная очистка данных при помощи регулярных выражений. Далее классифицируем полученные данные по лексемам и очищаем данные алгоритмом Левенштейна [6], если разница в один изменяемый символ. После этого происходит очистка по лексемам, по схожести и занесение очищенных данных в базу. Для каждого очищенного аудио исполнителя строится связь с пользователем.

Так же был спроектирован общий алгоритм работы системы, представленный на рисунке 2.

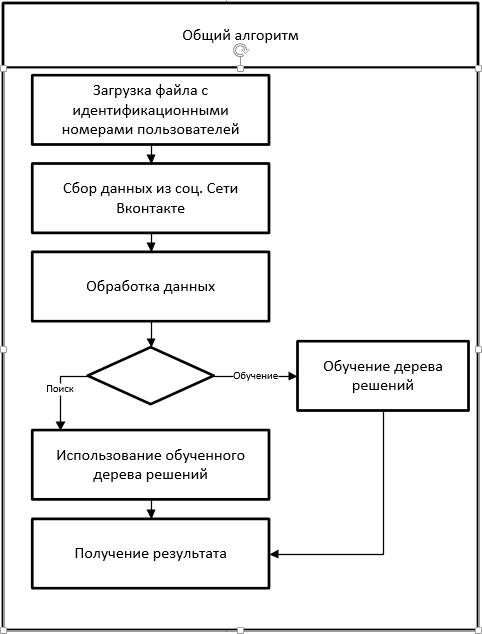


Рисунок 2 - Общий алгоритм работы программы

Первым делом в программу загружается файл с идентификационными номерами пользователей Вконтакте. Происходит сбор данных из Вконтакте, но без визуального осмотра. Если стоит задача обучить систему, то происходит переобучение по занесенным данным. Если стоит задача поиска, то используется уже готовое дерево решений и получаем результат работы программы.

**Алгоритм создания матрицы для обучения:**

Получаем данные из БД по всем пользователям для обучения у кого категория факультета удовлетворяет запросу.

Получаем данные из БД по всем музыкальным исполнителям, которые встречаются больше чем у 10 пользователей, отрезаем верхушку выборки свыше 1000. Полученные данные записываем в файл для прогнозирования.

Для каждого пользователя из БД получаем личный список музыки.

Проверяем встречается ли музыка пользователя во всей музыке, если встречается, то заменяем поле на 1 в строке пользователя. Если у пользователя нет ни одного аудио исполнителя из выборки, но не записываем его в таблицу.

В последний столбец матрицы вставляем категорию пользователя.

Полученную матрицу записываем в файл для дальнейшего использования.

Алгоритм представлен на рисунке.

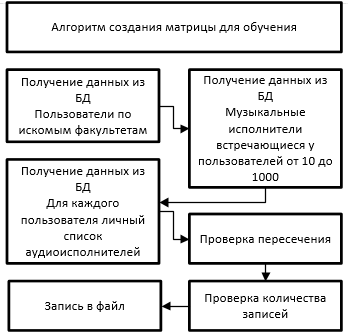


Рисунок – алгоритм составления матрицы

**Алгоритм обучения дерева решений:**

Считываем частями файл с матрицей для обучения.

Из матрицы получаем значения для обучения и категории. Разделяем выборку на обучение и тестирование. Перекрестной проверкой (Кросс-валидацией) разбиваем данные на 5 перемешанных частей. Обучаем дерево решений с глубиной до 30.

Получаем численную оценку качества модели, а именно долю по которым модель приняла правильное решение (Accuracy[http://bazhenov.me/blog/2012/07/21/classification-performance-evaluation.html]). Так же получаем F-меру – средневзвешенное между полнотой и точностью.

Записываем обученного дерева в файл для дальнейшего использования.

Алгоритм представлен на рисунке.

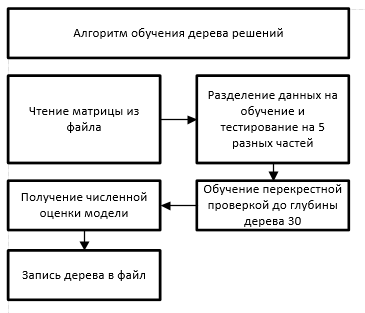


Рисунок – алгоритм

## Проектирование базы данных сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

Модель сущность-связь (англ. entity-relationship model, ERM, ER-модель) позволяет описывать концептуальные схемы предметной области.

ER-модель используется для высокоуровневого проектирования баз данных. С её помощью можно выделить ключевые сущности и обозначить связи, которые могут устанавливаться между этими сущностями.

ER-диаграмма представляет графическую структуру данных проектируемой БД. Сущности отображаются при помощи прямоугольников, таблиц, содержащих имя сущности - таблицы БД. Взаимосвязи сущностей отображаются в виде линий, соединяющих отдельные сущности.

Взаимосвязь показывает, что данные одной сущности ссылаются или связаны с данными другой.

В ходе проектирования базы данных была составлена ER - модель, рисунок # описание сущностей описано ниже.

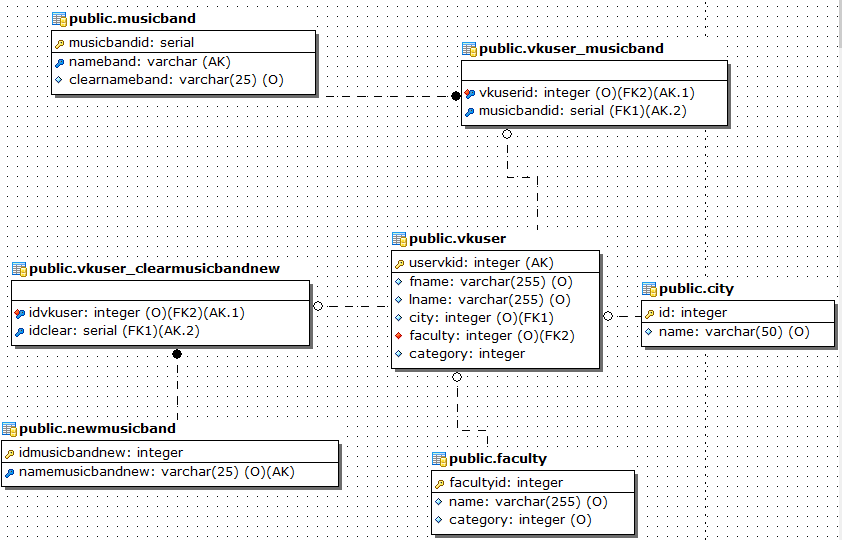


Рисунок # ER - модель.

Описание таблиц БД:

vkuser - таблица пользователя Вконтакте:

* uservkid – идентификационный номер пользователя Вконтакте
* fname – имя пользователя Вконтакет
* lname – фамилия пользователя Вконтакте
* city- - идентификационный номер города пользователя Вконтакте
* faculty - идентификационный номер факультета пользователя Вконтакте
* category – категория пользователя – обучающая или поисковая.

musicband – таблица аудио исполнителей из Вконтате:

* musicbandid - идентификационный номер аудио исполнителей Вконтакте
* nameband - название очищенного аудио исполнителя Вконтакте
* clearnameband – пред очищенные данные аудио исполнителя Вконтакте

vkuser\_clearmusicband – таблица реализующая связь многие ко многим:

* vkuserid – идентификационный номер пользователя Вконтакте
* musicbandid – идентификационный номер аудио исполнителей Вконтакте

newmusicband – таблица очищенных аудио исполнителей:

* idmusicband - идентификационный номер очищенного аудио исполнителей Вконтакте
* namemusicbandnew - название очищенного аудио исполнителя Вконтакте
* tag – поле для жанра музыки исполнителя из сервиса last.fm

vkuser\_clearmusicbandnew – таблица реализующая связь многие ко многим:

* idvkuser – идентификационный номер пользователя Вконтакте
* idclear – идентификационный номер очищенного аудио исполнителей Вконтакте

city - таблица описания городов:

* id – идентификационный номер города
* name – наименование города

facukty - таблица факультетов:

* facultyid - идентификационный номер факультета пользователя Вкотнакте
* name - название факультета
* category – категория распределения факультета

## Проектирование физического представления сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

Форма физического представления программной системы - это диаграмма развертывания (размещения). Необходима для полного физического представления системы. Диаграмма развертывания представлена на рисунке #.

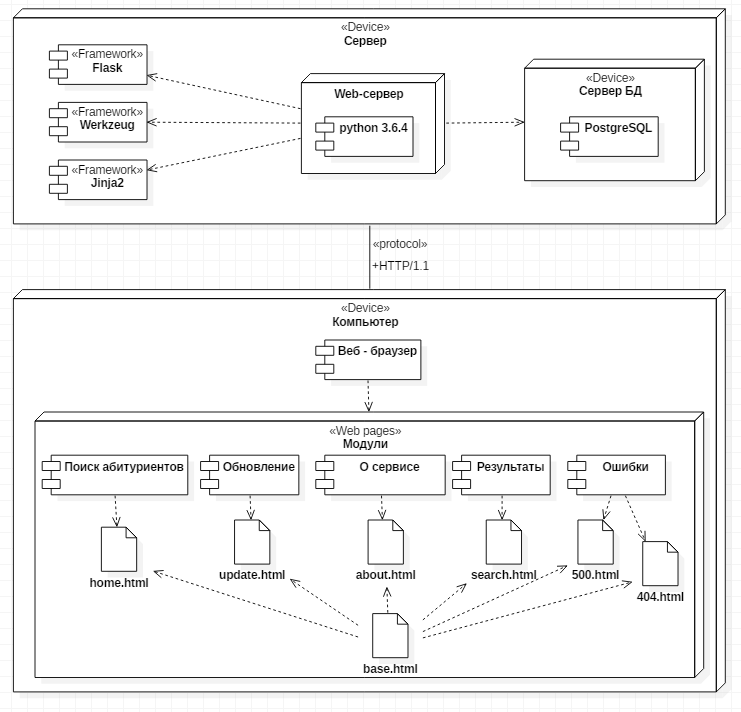


Рисунок # - диаграмма развертывания.

Диаграмма развертывания применяется для представления общей конфигурации и топологии распределенной программной системы и содержит изображение размещения компонентов по отдельным узлам системы. Кроме того, диаграмма развертывания показывает наличие физических соединений - маршрутов передачи информации между аппаратными устройствами, задействованными в реализации системы [#].

Клиент-серверная архитектура разделяет всю работающую систему на три различные части, определенным образом взаимодействующие между собой:

* клиентская приложение - это программа, работающая на компьютере пользователя и обеспечивающая интерактивное взаимодействие системы поиска абитуриентов с пользователем;
* серверная python - обеспечивает взаимодействие между пользователями и системой управления базами данных в клиент-серверном варианте работы.
* сервер базы данных - специализированная программа (чаще комплекс программ), предназначенная для организации и ведения базы данных.

Программа, работающая у пользователя, (клиентское часть) взаимодействует с сервером python, а она, при необходимости, обращается к серверу баз данных.

При этом физически сервер python и сервер баз данных могут располагаться как на одном компьютере, так и на разных. Это позволяет администратору при необходимости распределять нагрузку между серверами.

Для передачи данных между клиентскими приложениями и сервером, используется HTTP/1.1.

# Разработка сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

## Выбор языка программирования сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

Основной задачей ВКР является выявление закономерностей среди абитуриентов при помощи машинного обучения. На сегодняшний день, это направление отлично развито в высокоуровневом языке программирования Python. Данный ЯП имеет много вспоминающих библиотек для машинного обучения, поддерживается как front-end, так и back-end разработка. Для выполнения поставленных задач python наилучшее решение.

Для создания веб-приложения был использован микро-фреймворк Flask. Быстрый, простой, не имеет ни чего лишнего, нагружающего серверное оборудование, использует шаблонизатор Jinja2 и набор инструментов библиотеки Werkzeug для HTTP запросов.

Так как API VK AUDIO закрыты, то необходим сборщик данных. Для данного был выбран популярный инструмент для автоматизации браузера - Selenium.

Преимущества всего стека-технологий – бесплатно распространяемое ПО, большой объем информации в сети Интернет.

## Выбор СУБД для сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

База данных - совокупность взаимосвязанных данных, организованных в соответствии со схемой базы данных таким образом, чтобы с ними мог работать пользователь. [# ГОСТ 34.321-96]

Система управления базами данных(СУБД) - Совокупность программ и языковых средств, предназначенных для управления данными в базе данных, ведения базы данных и обеспечения взаимодействия ее с прикладными программами. [# ГОСТ 20886-85]

Реляционная база данных представляет собой множество взаимосвязанных таблиц, каждая из которых содержит информацию об объектах определенного вида. Каждая строка таблицы содержит данные об одном объекте, а столбцы таблицы содержат различные характеристики этих объектов – атрибуты [# Академия Microsoft: Технология Microsoft ADO .NET Ч.А. Кариев]

Выбор систему управления баз данных (СУБД) представляет сложную задачу. Один из важнейших этапов разработки баз данных. Выбранный программный продукт должен удовлетворять многим критериям: текущим потребностям, будущим потребностям, финансовым затратам на приобретение оборудования, программного обеспечения и его использования.

При разработке тестировались схожие СУБД: MySQL, Oracle, PostgreSQL.

Преимущества и недостатки MySQL:

К преимуществам относится: высокая безопасность, гибкость и простота в обращении, много готовых решений. К недостаткам относится: частые сбои при работе с одновременными операциями, малый функционал обработки текста.

Oracle и PostgreSQL продвинутые СУБД, с меньшим количеством недостатков, чем MySQL, способные расширяться за счет написания своих функций, огромный список возможностей, огромные сообщества в сети Интернет. Главный минус Oracle – эго цена. Поэтому в целях написания ВКР использовалась PostgreSQL.

Сервис поиска абитуриентов будет работать на СУБД PostgerSQL. Поэтому в рамках выпускной квалификационной работы будет использоваться она. Графические оболочки для работы с СУБД PostreSQL: HeidiSQL и pgAdmin 4. Данный выбор заключается в том, что они хорошо дополняют друг друга. PgAdmin 4 хорош для создания самой базы данных и ее настройки. HeidiSQL удобен для резервирования БД, просмотра и редактирования данных. Так же был рассмотрена графическая оболочка dbForge Studio Express for PostgreSQL. Данный продукт находится на стадии разработки и функционал продукта очень скуден, на момент написания ВКР был возможет лишь просмотр данных. Проектирование БД проводилось в сервисе dbdesigner.net и SQL Manager for PostgreSQL.

## Выбор модели для обучения сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

Сервис предполагает прогнозирование пользователей по нескольким классификациям. Деревья принятия решений – одна из моделей выполняющая поставленную задачу [http://scikit-learn.org/stable/modules/tree.html

https://basegroup.ru/community/articles/description]. Цель состоит в том, чтобы создать модель, которая предсказывает значение целевой переменной путем изучения простых правил принятия решений, полученных из данных. Контролируемый метод обучения. Простота в использовании. Возможность визуализации. Поддержка категориальных и числовых признаков. Для построения дерева решений использовалась библиотека Python для машинного обучения scikit-learn.

Для проверки дерева решений используют перекрестную проверку(Кросс-валидацию). Кросс-валидация - метод оценки аналитической модели и её поведения на независимых данных [http://datascientist.one/cross-validation/]. Использовалась кросс-валидация по K блокам (K-fold cross-validation). В этом случае исходный набор данных разбивается на K одинаковых по размеру блока. Из K блоков один оставляется для тестирования модели, а остающиеся K-1 блока используются как тренировочный набор. Процесс повторяется K раз, и каждый из блоков используется один раз как тестовый набор. Получаются K результатов, по одному на каждый блок, они усредняются. Преимущество такого способа перед случайным сэмплированием (random subsampling) в том, что все наблюдения используются и для тренировки, и для тестирования модели, и каждое наблюдение используется для тестирования в точности один раз.

## Разработка интерфейса сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

В ходе написания выпускной квалификационной работы был разработан минимальный интерфейс сервиса при помощи микро-фреймворка – Flask.

В сервисе присутствует три вкладки, «Главная» (Поиск абитуриентов), «О сайте», «Обновить». Интерфейс представлен на рисунках 5,6 ниже.

Описание страницы поиск абитуриентов: На странице расположены две кнопки – запуск поиска и выбор файла из файловой системы пользователя.

Описание страницы информации абитуриентов: На странице расположено описание сервиса, и возможная работа с ним.

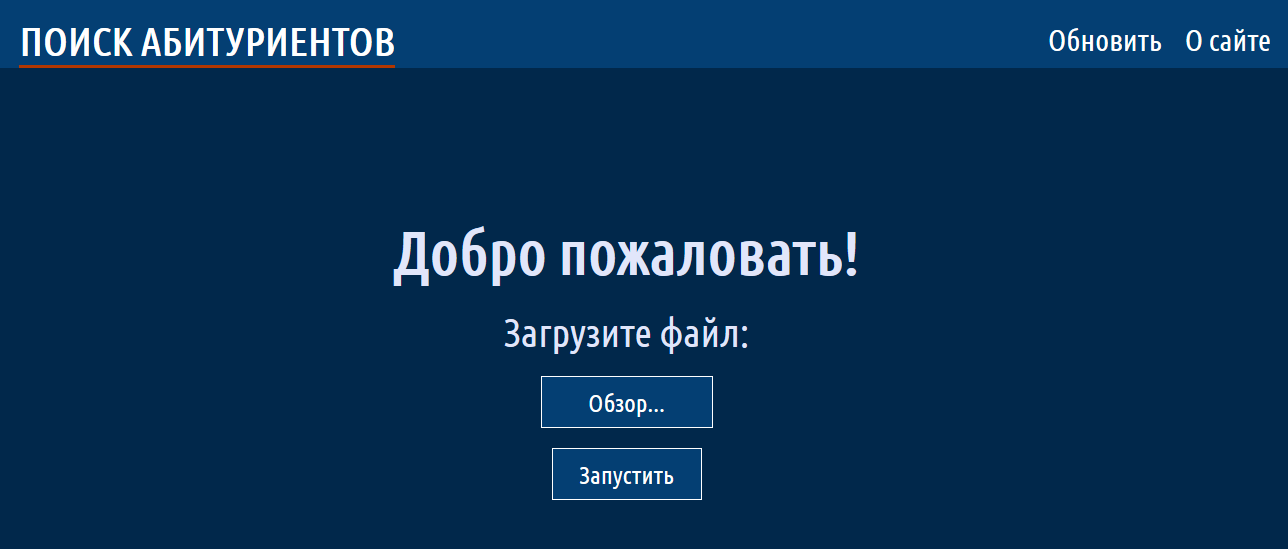


Рисунок 5 – Главная страница

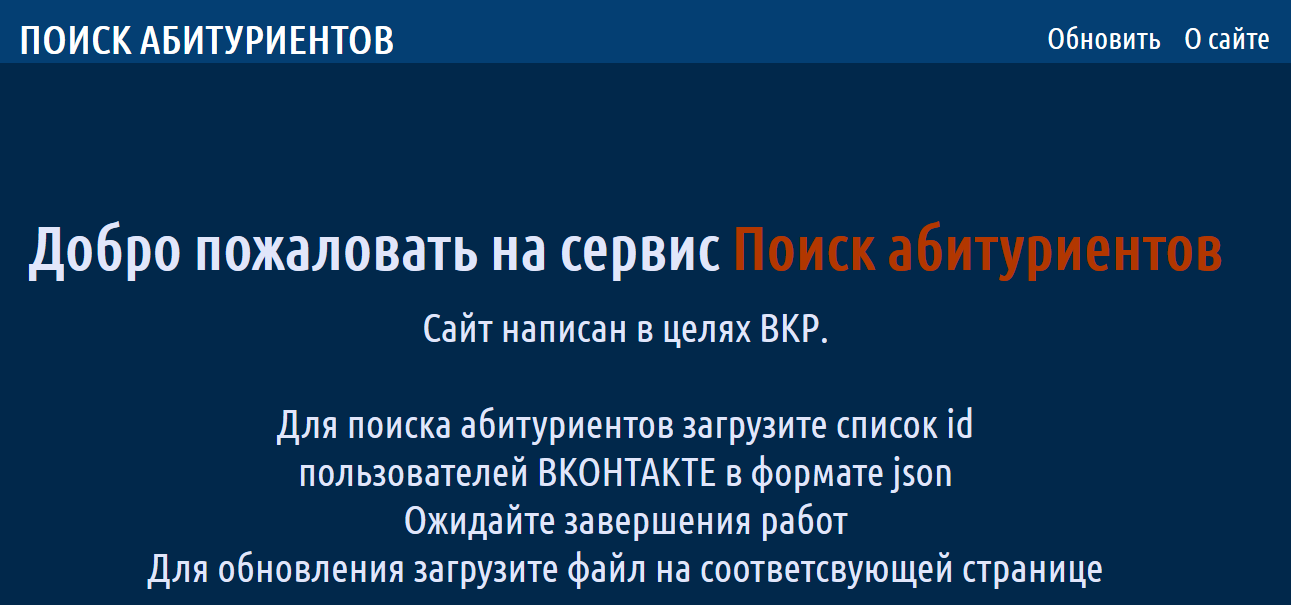


Рисунок 6 – Стреница о сервисе

## Разработка модели обучения сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

При разработке использовался ранее указанный алгоритм обучения. Код представлен на рисунке ниже.

Рисунок

## Разработка сборщика матрицы для обучения сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

При разработке использовался ранее указанный алгоритм сбора матицы для обучения. Код представлен на рисунке ниже.

Рисунок

## Разработка сборщика пользовательских данных сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

При разработке использовался ранее указанный алгоритм сбора матицы для обучения. Код представлен на рисунке ниже.

Рисунок

## Разработка очистки музыкальных исполнителей сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

## Разработка разбиения на категории факультетов сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

# Тестирование сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

## Тестирование интерфейса сервиса поиска абитуриентов на базе технологий NLP и машинного обучения фокус-группой

Для фокус-группы из 20 человек была составлена анкета с вопросами, (рисунок ниже#) и прототип интерфейса (рисунок ранее#). Фокус-группа состояла из разновозрастных участников с разным родом деятельности. Результаты представлены в таблице #.

Таблица 1 – Анкета для фокус-группы.

|  |  |
| --- | --- |
| **Вопрос** | **Ответ** |
| 1. Оцените читаемость текстов в программе по 10-балльной шкале.   (*1-очень низкая, 10-очень высокая*) |  |
| 1. Оцените шрифт, используемый в программном продукте. (*Да/Нет*) |  |
| 1. Оцените цветовую гамму программного продукта по 10 –балльной шкале. (*1-очень низкая, 10-очень высокая*) |  |
| 1. Насколько интуитивно понятна навигация в программе?  * *Совсем непонятна* * *Немного непонятна* * *Достаточно понятна* * *Хорошо понятна* |  |
| 1. Что бы вы поменяли в программном продукте? (*письменно*) |  |
| 1. Что нужно сделать, чтобы начать поиск абитуриентов? (*письменно*) |  |
| 1. Опишите не удобное расположение кнопок на формах (*письменно*) |  |
| 1. Что вам не нравится в нашем продукте? (*письменно*) |  |

Результаты анализа полученных анкет и решение на их основе:

Таблица # –Результаты оценки анкет фокус-группы.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Вопрос | Результат | Решение на основе результатов. |
| 1 | Оцените читаемость текстов в программе по 10-балльной шкале. (*1-очень низкая, 10-очень высокая*) | 4,8 | Требуются изменения. |
| 2 | Оцените шрифт, используемый в программном продукте. (*Да/Нет*) | Да | Изменений не требуется. |
| 3 | Оцените цветовую гамму программного продукта по 10 –балльной шкале. (*1-очень низкая, 10-очень высокая*) | 8,2 | Изменений не требуется. |
| 4 | Насколько интуитивно понятна навигация в программе?   * *Совсем непонятна* * *Немного непонятна* * *Достаточно понятна* * *Хорошо понятна* | Хорошо понятна | Изменений не требуется. |
| 5 | Что бы вы поменяли в программном продукте? (*письменно*) | Изменить толщину шрифта, шрифт не серьезный. | |
| 6 | Что нужно сделать, чтобы начать поиск абитуриентов? (*письменно*) | 1. Выбрать файл 2. Нажать кнопку запуска 3. Ждать завершения процесса. | |
| 7 | Опишите не удобное расположение кнопок на формах (*письменно*) | Кнопки расположены интуитивно, но слишком сжато друг к другу. | |
| 8 | Что вам не нравится в нашем продукте? (*письменно*) | 1) Уменьшить пустое пространство  2) Толщину шрифта  3) Шрифт слишком сложный для восприятия  4) Пустое расстояние на главной странице  5) Расстояние между кнопками | |

Исходя из результатов тестирования, программный продукт был хорошо принят фокус группой. На основе оценок и пожеланий выявлены проблемные участки сервиса, которые были исправлены.

# Заключение

# Список использованных источников

В. В. Бахтизин, Л. А. Глухова Технология разработки программного обеспечения Допущено Министерством образования Республики Беларусь в качестве учебного пособия для студентов

# Приложения

# Приложение А Техническое задание на создание сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

# Приложение Б Диаграмма вариантов использования

# Приложение В Диаграмма классов

# Приложение Г Диаграмма развёртывания

# Приложение Д Диаграмма базы данных сервиса поиска абитуриентов на базе технологий NLP и машинного обучения

ПРИЛОЖЕНИЕ #