****

Министерство науки и высшего образования Российской Федерации

**Федеральное государственное бюджетное**

**образовательное учреждение высшего образования «Казанский национальный исследовательский технологический университет» (ФГБОУ ВО «КНИТУ»)**

Кафедра «Интеллектуальных систем и управления информационными ресурсами»

Направление специальность 02.03.03 «Математическое обеспечение и администрирование информационных систем»

Тема выпускной квалификационной работы: «Рекомендательная система подбора университетов для абитуриентов»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

Заведующий кафедрой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

Руководитель проекта\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

Студент\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

Нормоконтролер\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

Казань 2022 г.

# З А Д А Н И Е

на курсовую работу (проект) студенту кафедры \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Тема работы (проекта): \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Исходные данные к проекту: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Содержание расчетно-пояснительной записки (включая перечень подлежащих разработке вопросов, включая вопросы стандартизации и контроля качества) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Перечень графического материала (схемной документации) \_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Консультанты по проекту (с указанием относящихся к ним разделов) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_г.

Руководитель проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

# ЛИСТ НОРМОКОНТРОЛЕРА

1. Лист является обязательным приложением к пояснительной записке дипломного (курсового) проекта.
2. Нормоконтролер имеет право возвращать документацию без рассмотрения в случаях:

- нарушения установленной комплектности,

- отсутствия обязательных подписей,

- нечеткого выполнения текстового и графического материала.

1. Устранение ошибок, указанных нормоконтролером, обязательно.

П е р е ч е н ь

замечаний и предложений нормоконтролера по дипломному (курсовому) проекту, студента

(группа, инициалы, фамилия)

|  |  |  |
| --- | --- | --- |
| Лист (страница) | Условное обозначение (код ошибок) | Содержание замечаний и предложений со ссылкой на нормативный документ, стандарт или типовую документацию |
|  |  |  |

Дата «\_\_\_\_» \_\_\_\_\_\_\_\_\_20\_\_\_г.

Нормоконтролер \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (инициалы, фамилия)

ОГЛАВЛЕНИЕ

[З А Д А Н И Е 2](#_Toc74292562)

[ЛИСТ НОРМОКОНТРОЛЕРА 2](#_Toc74292563)

[ВВЕДЕНИЕ 4](#_Toc74292564)

[1. РАЗРАБОТКА ПРОЕКТА ИНФОРМАЦИОННОЙ СИСТЕМЫ 5](#_Toc74292565)

[1.1. Исследование предметной области 5](#_Toc74292566)

[1.2. Цель и задачи 5](#_Toc74292567)

[1.3. Актуальность 6](#_Toc74292568)

[1.4. Глоссарий 6](#_Toc74292569)

[1.5. Функциональные требования 7](#_Toc74292570)

[1.6. Нефункциональные требования 7](#_Toc74292571)

[1.7. Существующие аналоги, их преимущества и недостатки 8](#_Toc74292572)

[1.7.1. «Поступи онлайн» 8](#_Toc74292573)

[1.7.2. «Вузотека» 9](#_Toc74292574)

[1.7.3. «Edunetwork» 10](#_Toc74292575)

[1.8. Макеты системы 11](#_Toc74292576)

[1.9. BPMN - модель 18](#_Toc74292577)

[1.10. Mind карта 19](#_Toc74292578)

[1.11. Создание базы данных 20](#_Toc74292579)

[1.11.1. Концептуальное проектирование 20](#_Toc74292580)

[1.11.2. Логическое проектирование 21](#_Toc74292581)

[1.11.3. Физическое проектирование 25](#_Toc74292582)

[1.12. Сбор данных 26](#_Toc74292583)

[1.13. Импорт данных в БД 31](#_Toc74292584)

[2. РАЗРАБОТКА ПРИЛОЖЕНИЯ 34](#_Toc74292585)

[2.1. Алгоритм определения направления путем прохождения проф. ориентационного теста 34](#_Toc74292586)

[2.2. Алгоритм формирования рекомендуемого списка университета 35](#_Toc74292587)

[2.3. Открытие проекта 35](#_Toc74292588)

[2.4. Обзор проекта 36](#_Toc74292589)

[2.4.1. Routes 37](#_Toc74292590)

[2.1.1. Routes.js 38](#_Toc74292591)

[2.1.2. Main 39](#_Toc74292592)

[2.1.3. Test 41](#_Toc74292593)

[2.1.4. LearningProfiles 42](#_Toc74292594)

[5. ЗАКЛЮЧЕНИЕ 48](#_Toc74292595)

[6. СПИСОК ЛИТЕРАТУРЫ 49](#_Toc74292596)

# ВВЕДЕНИЕ

В последнее время комиксы становятся все популярнее в мире, а люди все чаще используют веб-приложения для их чтения. Также очень популярны комиксы, выходящие на иностранных языках, однако перевод произведений на родной язык выходит через некоторое время. Чтобы людям не приходилось ждать перевод, было создано веб-приложение, позволяющее читать комикс и распознавать текст с изображения, что позволит в дальнейшем перевести его на родной язык с помощью машинного перевода.

Цель данной работы – разработать веб-приложение для чтения комиксов и реализовать функцию распознавания текста с изображений.

Для достижения этой цели необходимо выполнить следующий перечень задач:

1. Анализ предметной области;
2. Определение функциональных и нефункциональных требований к системе;
3. Построение макетов системы;
4. Создание базы данных;
5. Разработка клиентской и серверной частей веб-приложения.

# РАЗРАБОТКА ПРОЕКТА ИНФОРМАЦИОННОЙ СИСТЕМЫ

## Анализ предметной области

Для людей, читающих комиксы, необходимо приложение, позволяющее

Для абитуриентов и школьников необходим портал, в котором был бы представлен перечень престижных высших учебных заведений, городов для поступления, а также список специальностей и профилей обучения, которые характеризуются уникальным кодом, заведенным в общероссийский классификатор специальностей по образованию, для всех вышеуказанных разделов предусмотреть фильтрацию данных по заданному критерию.

На портале представить профессионально - ориентационный тест, который позволит определить наиболее близкий для абитуриента профиль обучения по прохождению теста, в случае, если он еще не определился с направлением.

По введенным результатам сдачи Единого государственного экзамена по трем предметам для абитуриента необходимо подобрать перечень высших учебных заведений (далее по тексту - ВУЗ), которые наиболее актуальны для поступления, список выводить в ранжированном порядке.

## Актуальность

Для каждого школьника, абитуриента выбор профессии, на обучение которой будет потрачено несколько лет, является очень ответственным шагом, поэтому к ее выбору необходимо подходить с полной осознанностью и рассудительностью [1]. В данному случае сервис по подбору университетов поможет абитуриенту с выбором, ведь проблема профессионального распределения в образовательные учреждения была актуальна всегда, а в связи с новыми образовательными реформами страны, связанными со сдачей Единых государственных экзаменов, она становится особенно значимой.

## Глоссарий

Таблица 1.1. - Глоссарий

|  |  |
| --- | --- |
| Термин, сокращение | Определение |
| BPMN | Business Process Model and Notation |
| FK | Внешний ключ |
| PK | Первичный ключ |
| SQL | Structured query language |
| JS | JavaScript |
| БД | База данных |
| Бэк | Серверная часть портала |
| ВУЗ | Высшее учебное заведение |
| ЕГЭ | Единый государственный экзамен |
| Инфо. графика | Информационная графика |
| Проф. ориентационный тест | Профессионально – ориентационный тест |
| СУБД | Система управления базами данных |
| Фронт | Клиентская часть портала |

## Функциональные требования

Функциональные требования – это перечень действий, сервисов, которые должна выполнять система, с описанием того, как система должна реагировать на тот или иной шаг пользователя, другими словами функциональные требования должны отвечать на вопрос: «Что должна делать система?» [3].

Для разрабатываемой системы определены следующие функциональные требования:

* наличие тестов, по прохождению которых определяется наиболее близкое направление для данного пользователя;
* краткий обзор специальностей, предоставляющий информацию о наименовании специальности, его коде;
* быстрый переход на сайт того или иного университета;
* возможность просмотра рекомендательного списка университетов для поступления по заданным баллам ЕГЭ/городу;
* обзор наиболее часто встречающихся вопросов у абитуриентов;
* фильтрация данных по заданному критерию.

## Нефункциональные требования

Нефункциональные требования – это перечень характеристик, которые должны быть присуще разрабатываемой системе, также они определяют пользовательские требования к системе и отвечают на вопрос «Какой должна быть система?»

В данном случае выделены следующие нефункциональные требования:

* легкость в навигации;
* легкость в использовании;
* ограничение между пользовательским интерфейсом и серверной части;
* «дружелюбность»;
* приятная цветовая гамма.

## Существующие аналоги, их преимущества и недостатки

### «Поступи онлайн»

Сайт для абитуриентов России, в котором представлена информация о высших учебных заведениях страны, справочная информация для поступления в ВУЗ, представлен перечень специальностей и профили образования [4]. Краткий обзор данного портала представлен на рисунке 1.1.

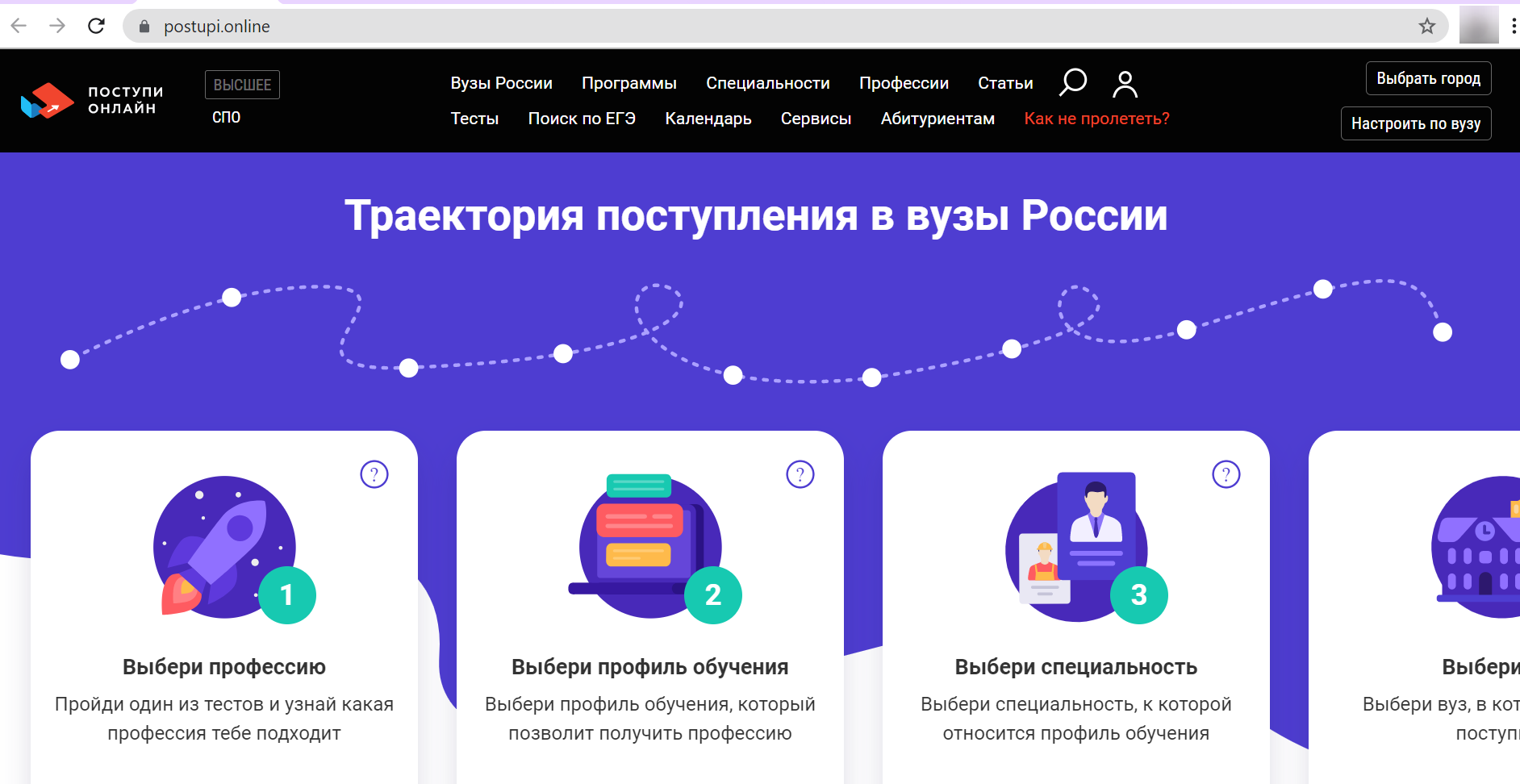


Рисунок 1.1. – Обзор сайта «Поступи онлайн».

Таблица 1.2 – Преимущества и недостатки системы «Поступи онлайн».

|  |  |
| --- | --- |
| Преимущества системы | Недостатки системы |
| 1. Наличие проф. ориентационных тестов, которые позволят определить профиль обучения, который близок именного данному пользователю; | 1. Проф. ориентационные тесты платные, либо доступны только по подписке на информационную рассылку от данного портала; |
| 1. Наличие краткого обзора специальностей; | 1. Нет возможности перейти на желаемый сайт ВУЗа с портала; |
| 1. Возможность просмотреть рекомендуемый список специальностей по заданным баллам ЕГЭ. | 1. Большинство разделов сайта заблокировано, доступно по предоплате. |

### «Вузотека»

Простой для использования сайт, в котором представлены списки престижных университетов Россииv[5]. Краткий обзор данного портала представлен на рисунке 1.2.

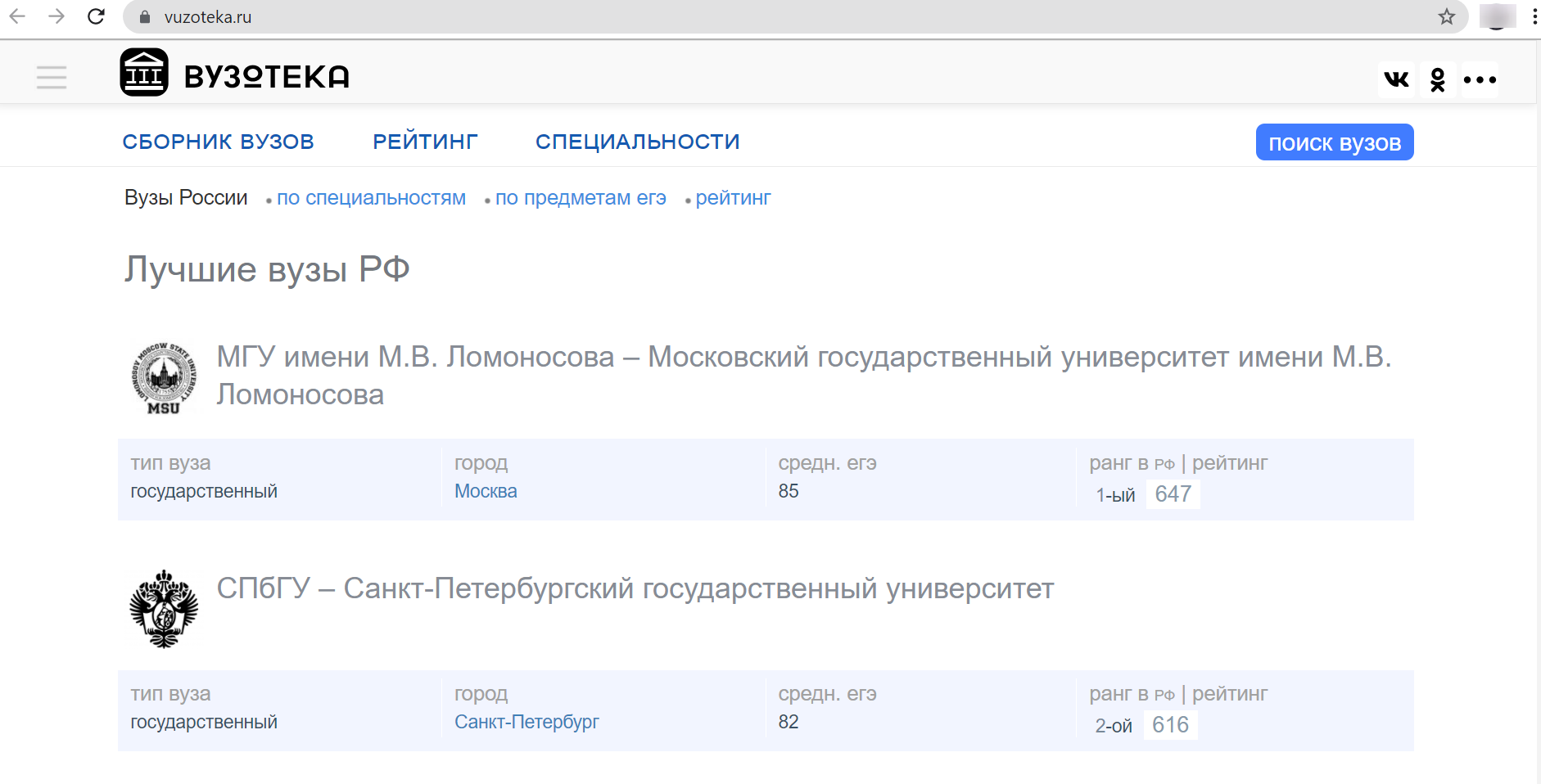


Рисунок 1.2. – Обзор сайта «Вузотека»

Таблица 1.3 – Преимущества и недостатки системы «Вузотека».

|  |  |
| --- | --- |
| Преимущества системы | Недостатки системы |
| 1. Наличие ТОП ВУЗов России, городов для поступления | 1. Отсутствует перечень профилей обучения |
| 1. Простой и удобный интерфейс | 1. Отсутствует справочный материал |
| 1. Удобная навигация по порталу | 1. Помимо рейтингов университетов, городов и т.д. не представлена другая информация |

### «Edunetwork»

Информационный портал «Edunetwork» предоставляет список учебных заведений России, Информацию по ВУЗам, а также краткий справочник абитуриента, в котором указана важная информация для поступления [6]. Краткий обзор данного портала представлен на рисунке 1.3.

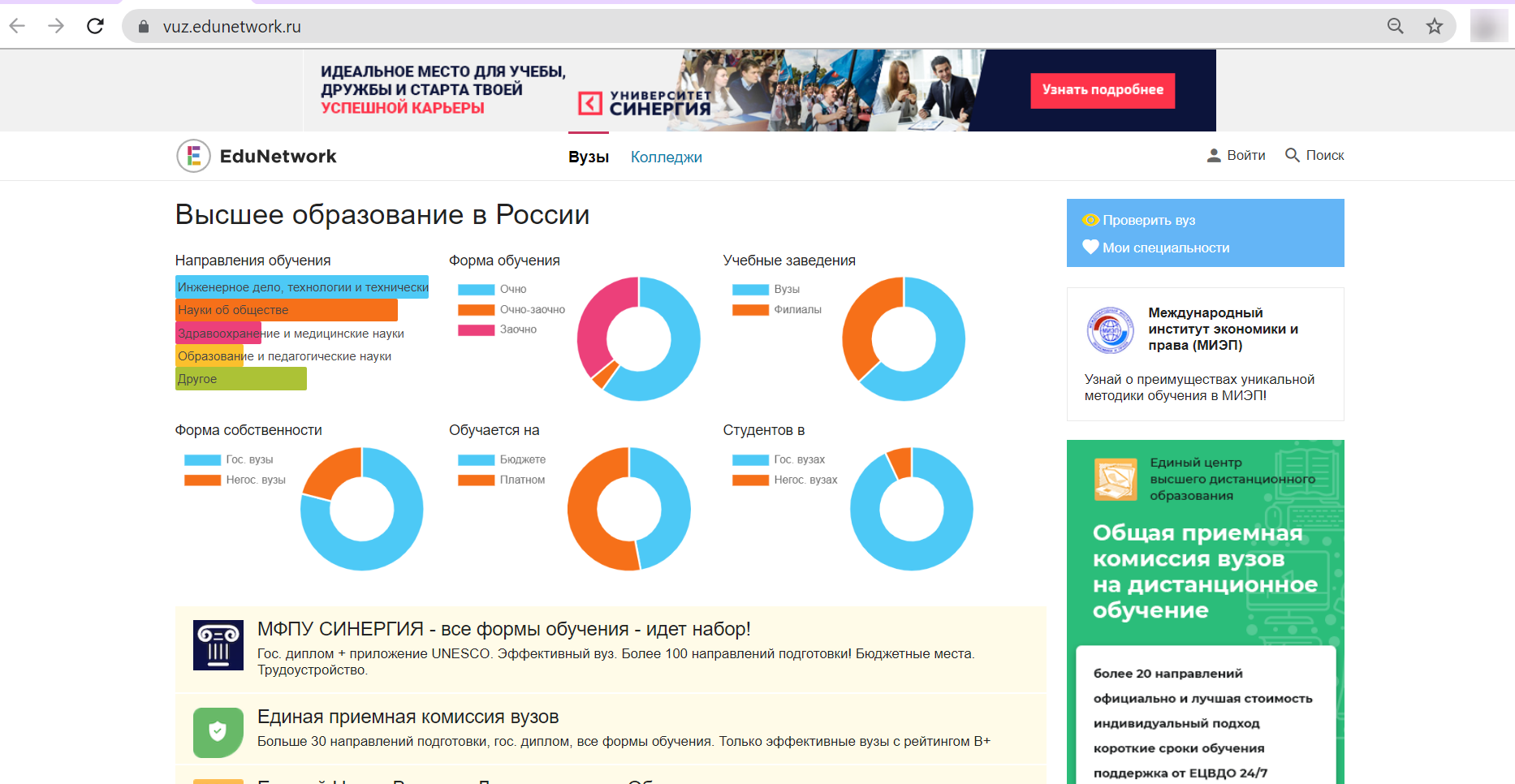


Рисунок 1.3. - Обзор сайта «Edunetwork».

Таблица 1.4 – Преимущества и недостатки системы «Edunetwork».

|  |  |
| --- | --- |
| Преимущества системы | Недостатки системы |
| 1. Динамическая инфо. графика, позволяющая проанализировать отображаемые данные; | 1. Отсутствие обзора профилей обучения; |
| 1. Наличие личного кабинета; | 1. Неудобная навигация; |
| 1. Наличие справочного материала. | 1. Достаточно яркое цветовое решение. |

## Макеты системы

Важно понимать отличие визуального образа, разрабатываемого с учетом технических возможностей HTML – верстки, от Wirefram, который представляет собой простой экскиз. Макет сайта – это практически весь дизайн. Он определяет порядок всех элементов ресурса, цветовые оттенки, формы, конкретное расположение компонентов по отношению друг к другу [7].

Создание макетов Системы проходило в онлайн – сервисе по разработке интерфейсов и прототипированию Figma.

Более детально каждый макет представлен на рисунках ниже:

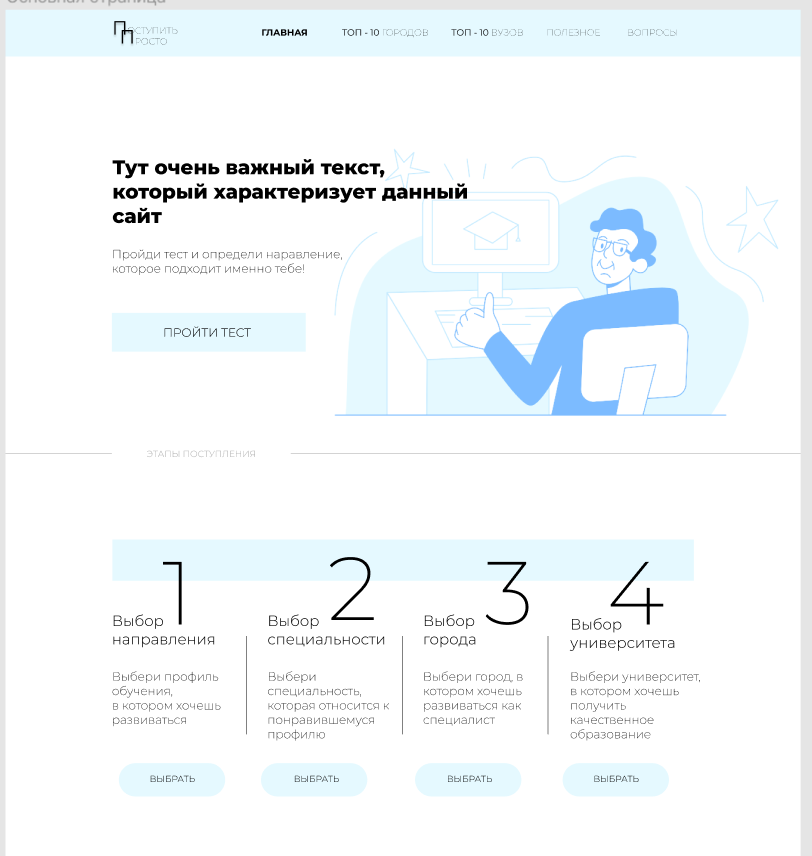


Рисунок 1.4. – Макет «Основная стрница и этапы поступления».

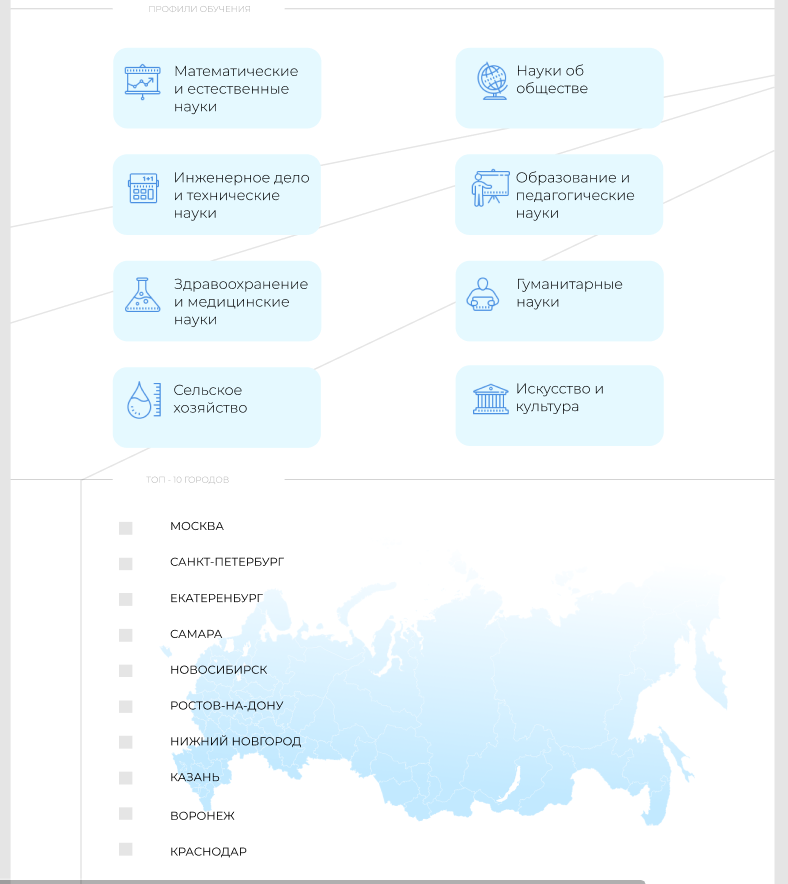


Рисунок 1.5. – «Профили обучения и топ – 10 городов».

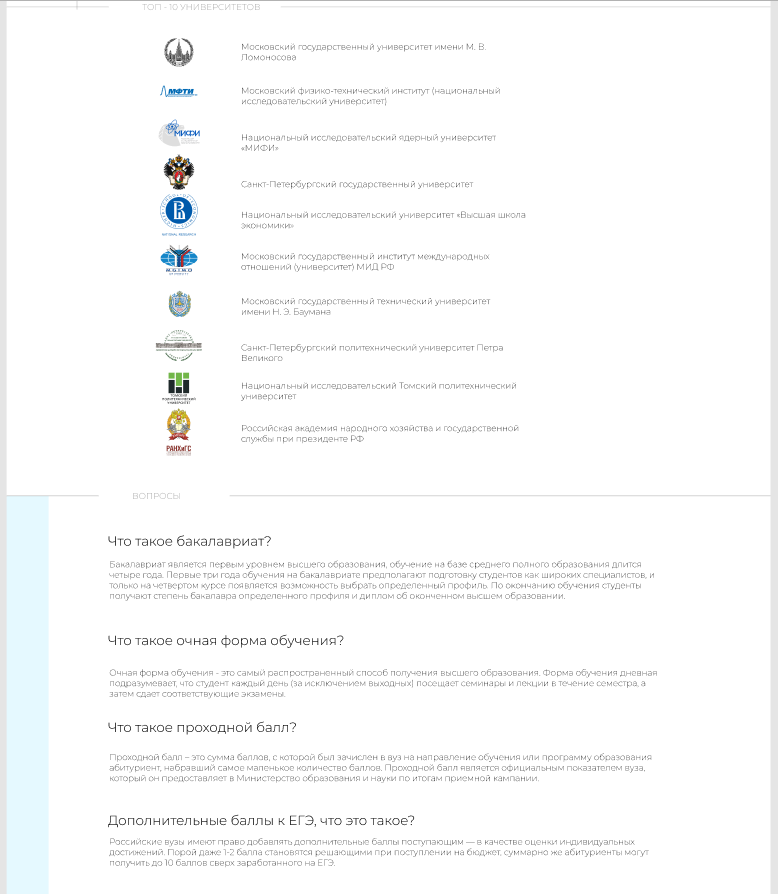


Рисунок 1.6. – «Топ – 10 университетов и вопросы».

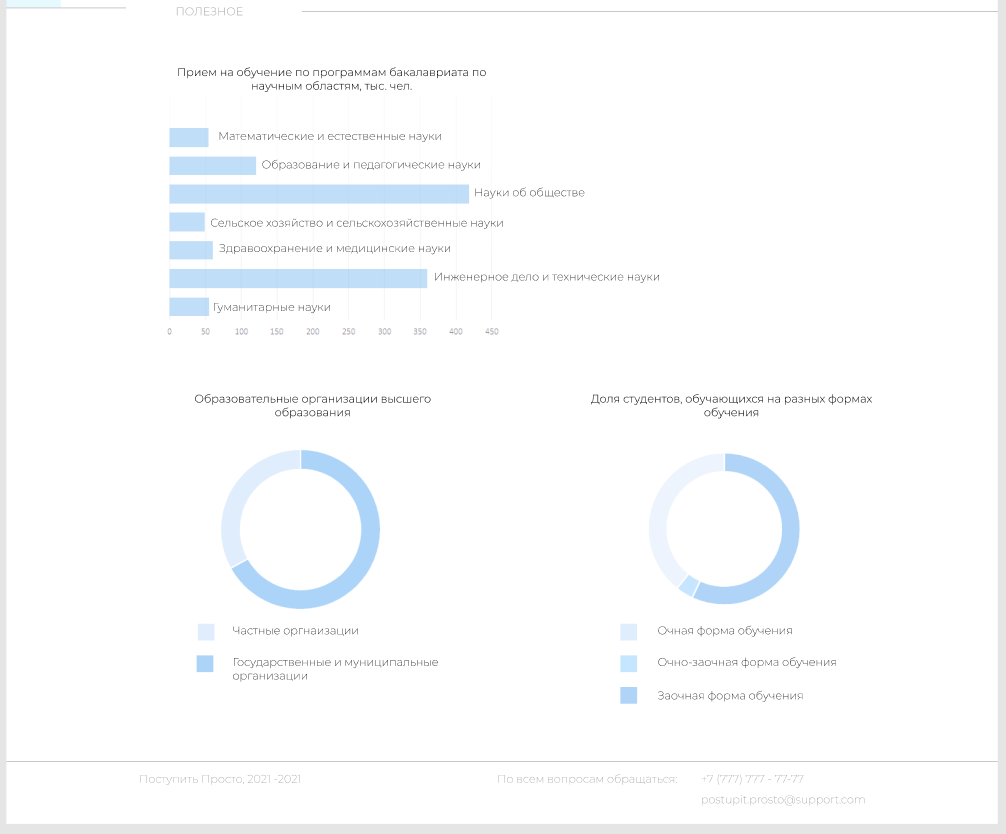


Рисунок 1.7. – «Полезное и footer».

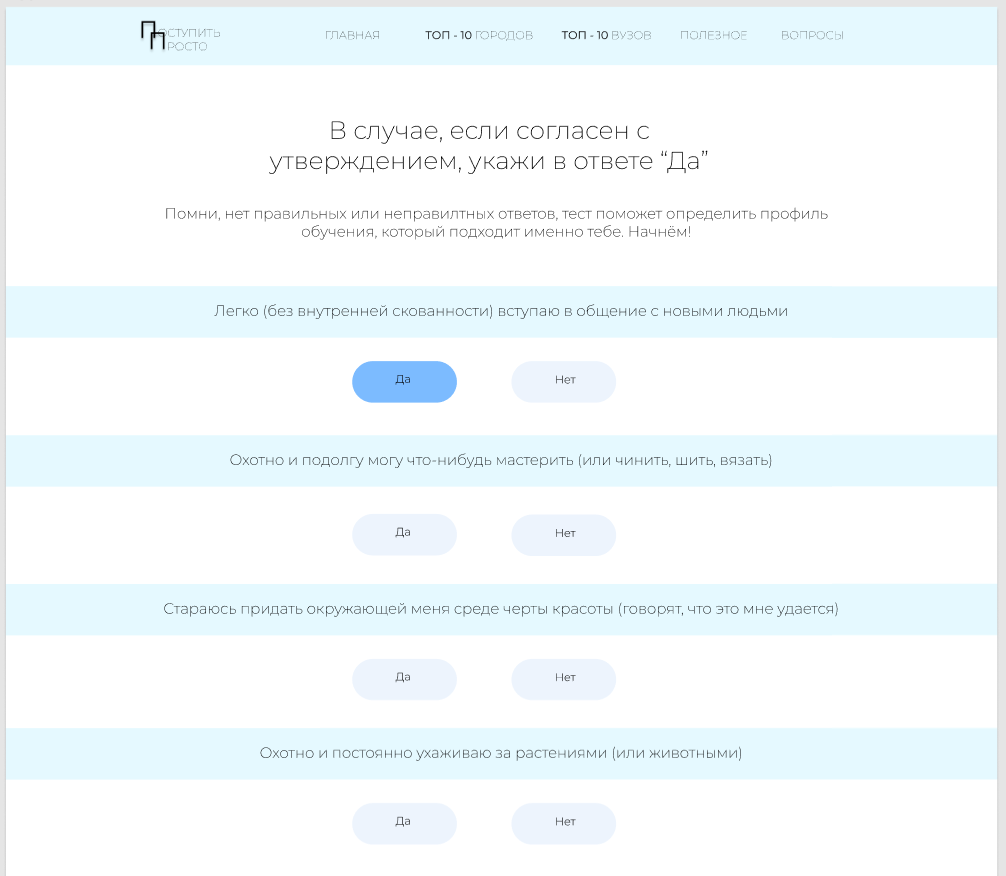


Рисунок 1.8. – «Тест».

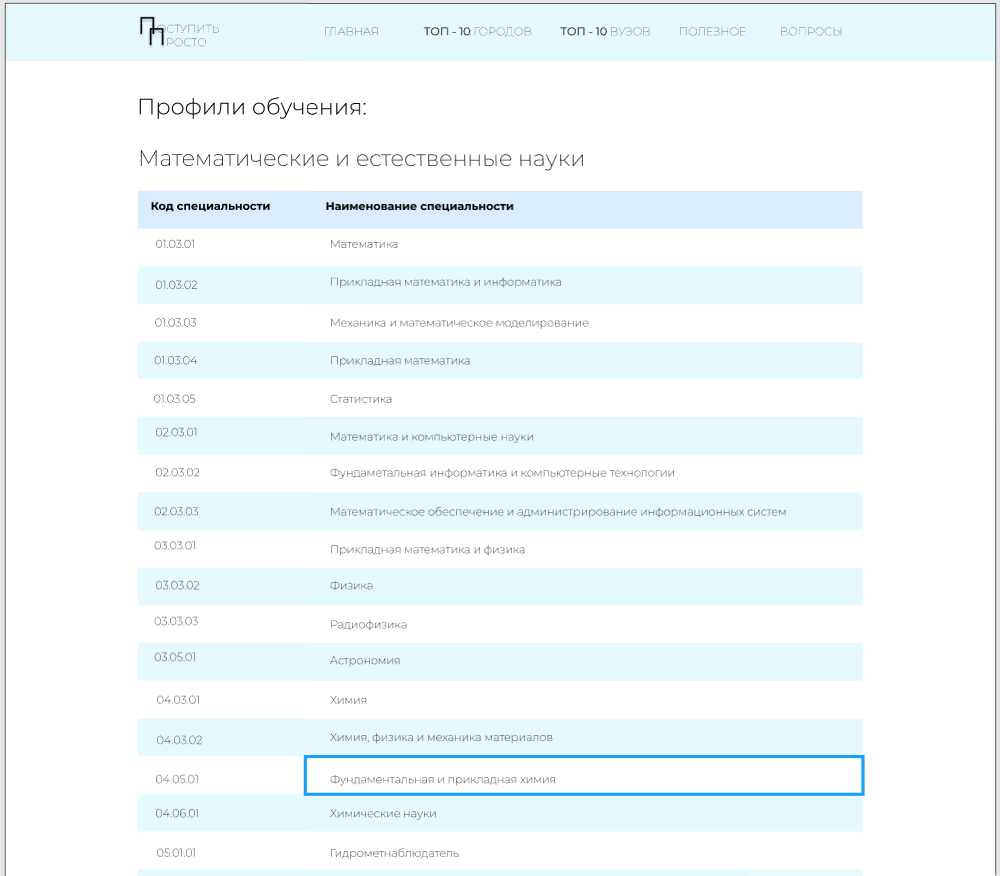


Рисунок 1.9. – «Реестр профилей обучения».

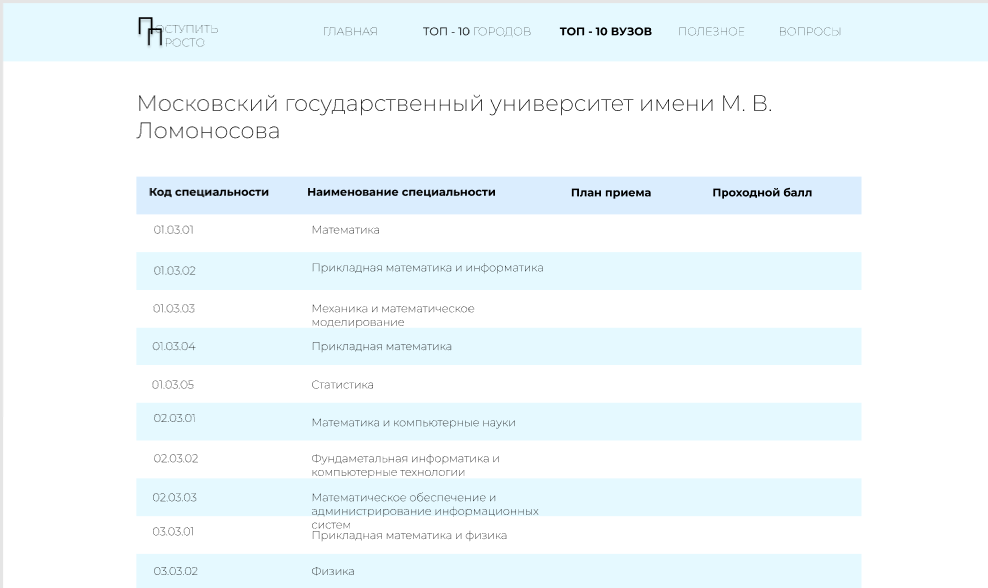


Рисунок 1.10. – «Реестр университетов».

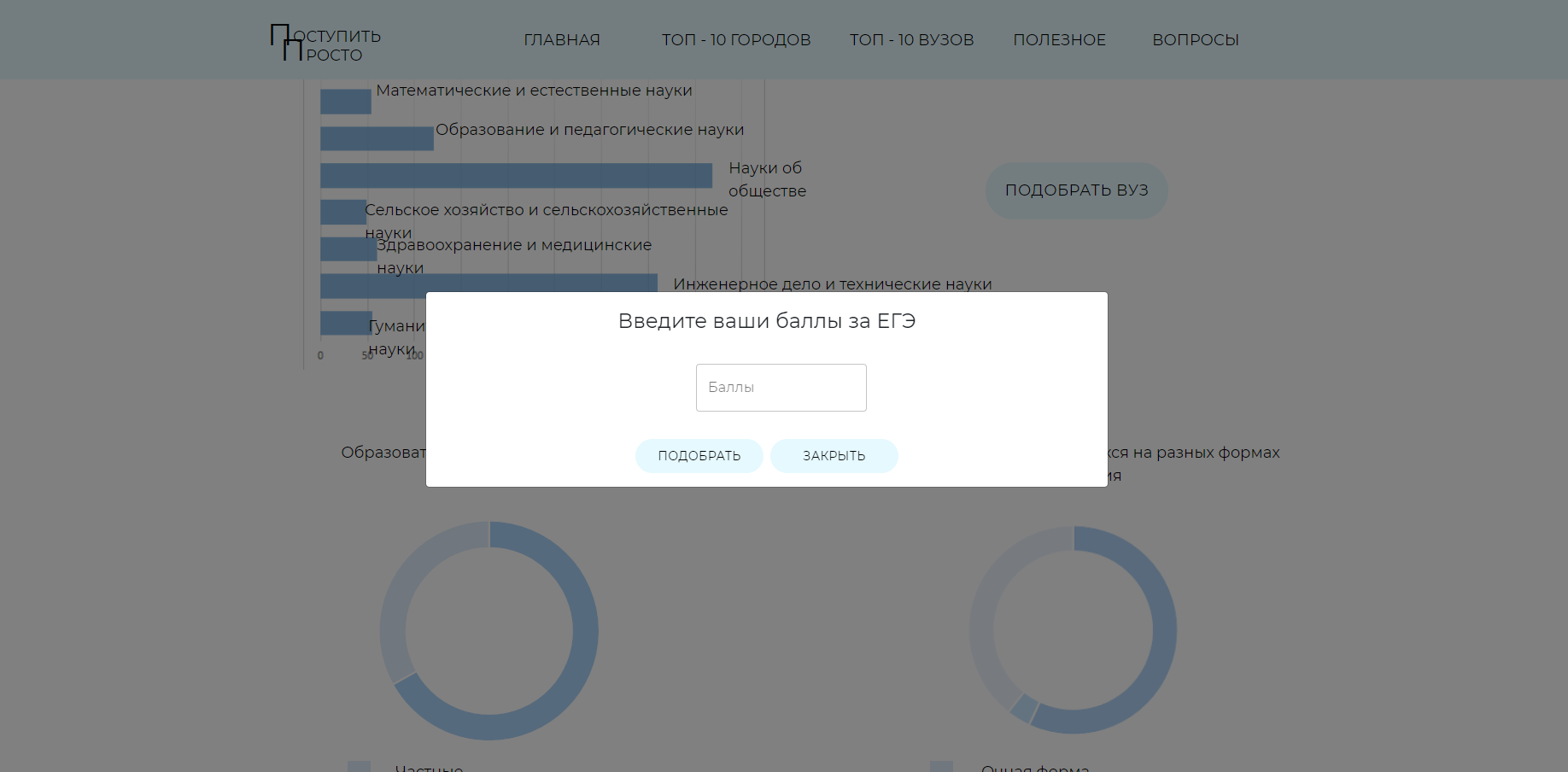


Рисунок 1.11 – «Модальное окно по определению рекомендательного списка ВУЗов для поступления».

## BPMN - модель

Спецификация BPMN описывает условные обозначения и их описание в XML для отображения бизнес-процессов в виде диаграмм бизнес-процессов. BPMN ориентирована как на технических специалистов, так и на бизнес-пользователей. Для этого язык использует базовый набор интуитивно понятных элементов, которые позволяют определять сложные семантические конструкции. Кроме того, спецификация BPMN определяет, как диаграммы, описывающие бизнес-процесс, могут быть трансформированы в исполняемые модели [8].

Далее на рисунке 1.12. представлено описание бизнес – процесса по выполнению возможных функций при посещении разрабатываемого портала.

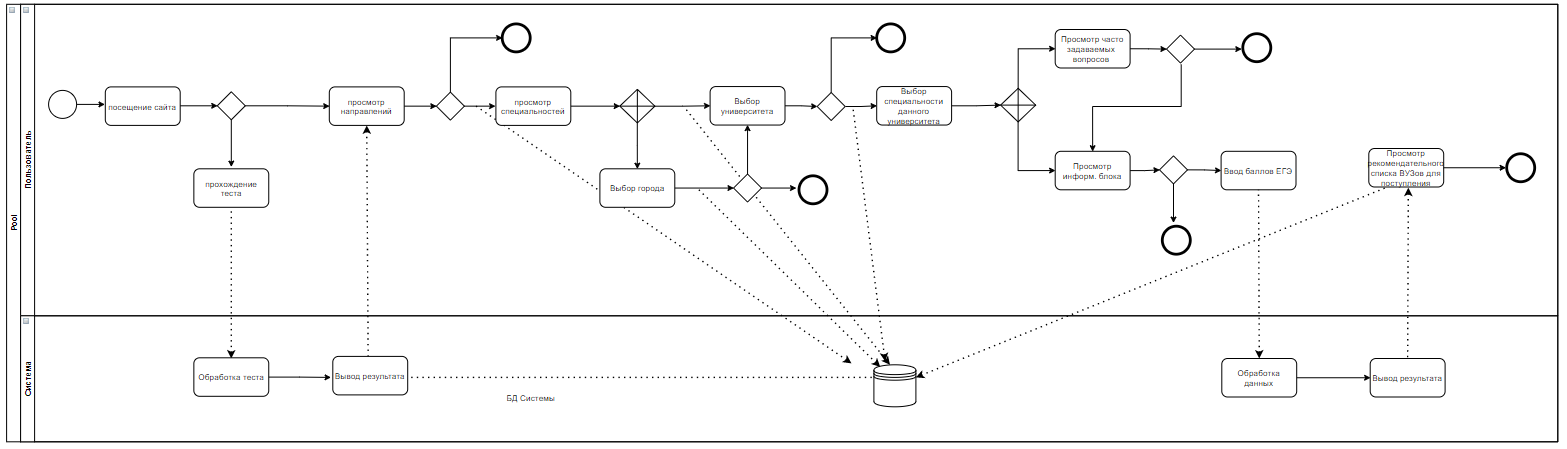


Рисунок 1.12. – BPMN модель

## Mind карта

Mind карта представляет собой некую «дорожную карту» сайта. На ней определено, что система должна делать, какие функции выполнять, какие разделы доступны и где они находятся. Подробное описание представлено ниже на рисунке 1.29.

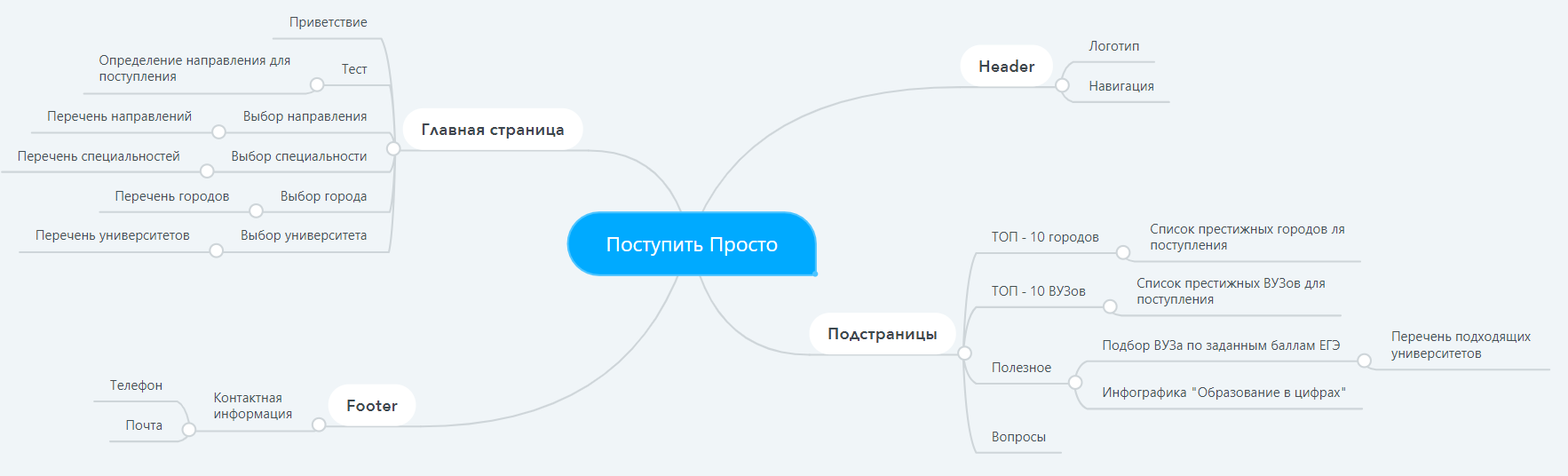


Рисунок 1.13. – Mind карта

На портале выделено 4 основных блока:

1. Главная страница – это основная страница портала. На ней представлено приветствие и возможность пройти на проф. ориентационный тест. Далее возможно просмотреть перечень специальностей и профилей;
2. Header – самая верхняя часть портала, на которой расположена панель навигация и логотип системы;
3. Footer – самая нижняя часть портала, на которой расположена контактная информация в виде номера телефона и почты;
4. Под страницы – это те страницы, на которые возможно перейти из панели навигации, с возможностью просмотра инфо. графики и подбора рекомендуемого перечня университетов.

## Создание базы данных

Данные Системы хранятся в БД, дальнейшая их обработка проходила в приложении pgAdmin с помощью СУБД PostgreSQL.

### Концептуальное проектирование

Концептуальное проектирование основывается на концептуальной модели – абстракции предметной области, которая базируется на графических диаграммах [9].

Популярной методикой создания концептуальной модели является методика построения модели «Сущность – связь».

Выделяются следующие базовые сущности выше упомянутой предметной области:

* Сущность «profile» содержит в себе информацию о доступных профилях обучения;
* Сущность «specialty» содержит в себе информацию о доступных специальностях;
* Сущность «university» содержит в себе информацию о доступных университетах;
* Сущность «city» содержит в себе информацию о доступных городах;

Для сущности «profile» определены следующие атрибуты:

1. profile\_code – определяет уникальный код профиля обучения;
2. profile\_title – определяет наименование профиля обучения.

Для сущности «specialty» определены следующие атрибуты:

1. specialty \_code – определяет уникальный код специальности;
2. specialty \_title – определяет наименование специальности.

Для сущности «university» определены следующие атрибуты:

1. university \_code – определяет уникальный код ВУЗа;
2. university \_title – определяет наименование ВУЗа.

Для сущности «city» определены следующие атрибуты:

1. city \_code – определяет уникальный код города;
2. city \_title – определяет наименование города.

### Логическое проектирование

Модель "сущность-связь" основывается на некоторой важной семантической информации о реальном мире и предназначена для логического представления данных. Она определяет значения данных в контексте их взаимосвязи с другими данными. Из модели "сущность-связь" могут быть порождены все существующие модели данных (иерархическая, сетевая, реляционная, объектная), поэтому она является наиболее общей. Представляет собой графическую модель, основанную на блоках и соединяющих их линиях, с помощью которых можно описывать объекты и отношения между ними [10]

В основе модели лежат следующие базовые понятия:

* Сущность - реальный или абстрактный объект, имеющий существенное значение для предметной области. Сущность должна иметь уникальное имя, обладать одним или несколькими атрибутами.
* Атрибут – характеристика сущности, предназначенная для определения ее свойств. Атрибут, который однозначно идентифицируется конкретным экземпляром сущности, называются ключевым.
* Связь – именованная ассоциация, которая устанавливается между несколькими сущностями.

Существует три вида связей между таблицами баз данных:

* Один к одному (1:1);
* Один ко многим (1 : М);
* Многие ко многим (М : М).

Связь (1 : 1) "один к одному" означает, что каждому экземпляру объекта А соответствует только один экземпляр объекта Б, и наоборот. В результате этой связи создается новая таблица.

Связь (1 : М) "один ко многим" означает, что каждому экземпляру объекта А может соответствовать несколько экземпляров объекта Б, а каждому экземпляру объекта Б только один экземпляр объекта А. В результате к сущности А добавляется первичный ключ сущности Б.

Связь (М : М) "многие ко многим" означает, что каждому экземпляру объекта А может соответствовать несколько экземпляров объекта Б, и наоборот. Итогом этой связи служит новая таблица с первичными ключами обеих сущностей.

В рассматриваемой базе данных ранее были выявлены следующие сущности:

1. Profile (profile \_code, profile \_title);
2. Specialty (specialty \_code, specialty \_title);
3. University (university \_code, university \_title);
4. City (city \_code, city \_title).

По сущностям базы данных «Выдача кредитов банком» можно сказать, что будущие таблицы будут иметь разные типы связей.

Определяются следующие типы связей между сущностями:

Profile – Specialty. Один и тот же профиль указан на нескольких специальностях, но в то же время одна специальность может определяться только по одному профилю. Отсюда тип связи «Profile – Specialty» один ко многим в соответствии с рисунком 1.12. В результате к сущности Specialty добавляется первичный ключ таблицы Profile - profile\_code.

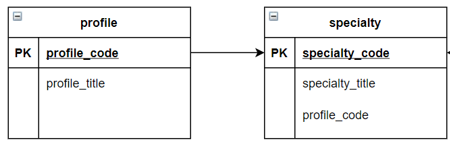


Рисунок 1.14. – Связь Profile – Specialty

University – Specialty. В одном и том же университете может быть представлено несколько специальностей, в то же время одна специальность может определяться в нескольких университетах. Отсюда тип связи «Profile – Specialty» многие ко многим в соответствии с рисунком 1.13. В результате создается новая сущность Univer\_Special.

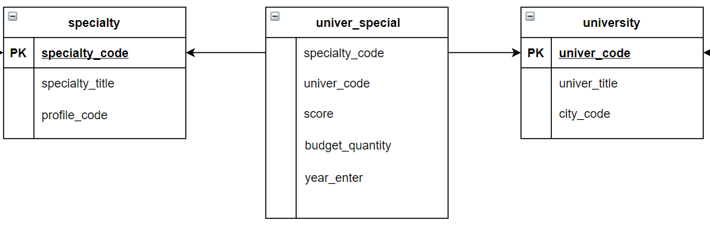


Рисунок 1.15. - Связь University – Specialty.

University – City. В одном и том же городе может быть представлено несколько университетов, в то же время один университет может находиться только в одном городе. Отсюда тип связи «University – City» один ко многим в соответствии с рисунком 1.14. В результате к сущности University добавляется первичный ключ таблицы City - сity \_code.

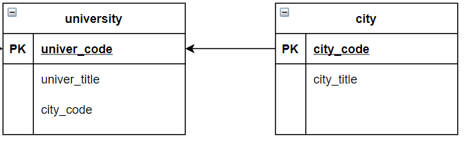


Рисунок 1.16. - Связь University – City.

Для поддержания БД в устойчивом состоянии используется ряд механизмов, которые получили обобщенное название средств поддержки целостности. Приведение структуры БД в соответствие этим ограничениям - это и есть нормализация [11].

Нормализация позволяет ввести ограничение на формирование таблиц, которые допускают удаление дублирования, обеспечивает непротиворечивость хранимых данных и уменьшает трудозатраты на ведение БД.

Процесс нормализации заключается в разложении исходных отношений базы данных на более простые отношения.

Процесс нормализации основан на понятии функциональной зависимости атрибутов: атрибут А зависит от атрибута Б, если в любой момент времени каждому значению атрибута Б соответствует не более одного значения атрибута А.

Понятие нормализация подразделяется на несколько нормальных норм (каждой нормальной форме соответствует свой набор ограничений).

В реляционных базах данных выделяют следующую последовательность нормальных форм:

1) Первая нормальная форма (1 НФ);

2) Вторая нормальная форма (2 НФ);

3) Третья нормальная форма (3 НФ);

4) Нормальная форма Бойса – Кодда (НФБК);

5) Четвертая нормальная форма (4 НФ);

6) Пятая нормальная форма (5 НФ).

Основное свойство нормальной формы: каждая последующая нормальная форма улучшает свойства предыдущей, при этом при переходе к следующей нормальной форме свойства предыдущей сохраняются.

Первая нормальная форма. Объект находится в первой нормальной форме, когда все его атрибуты имеют единственное значение. Если в каком – либо атрибуте есть повторяющееся значение, то из этого атрибута создается новое отношение.

Вторая нормальная форма. Отношение находится во второй нормальной форме, если оно уже находится в первой нормальной форме и если каждый его не ключевой атрибут зависит от уникального идентификатора (первичного ключа).

Третья нормальная форма. Отношение находится в третьей нормальной форме, если оно находится во второй нормальной форме и если не один его не ключевой атрибут не зависит от другого не ключевого атрибута.

Далее база данных будет приведена к третьей нормальной форме:

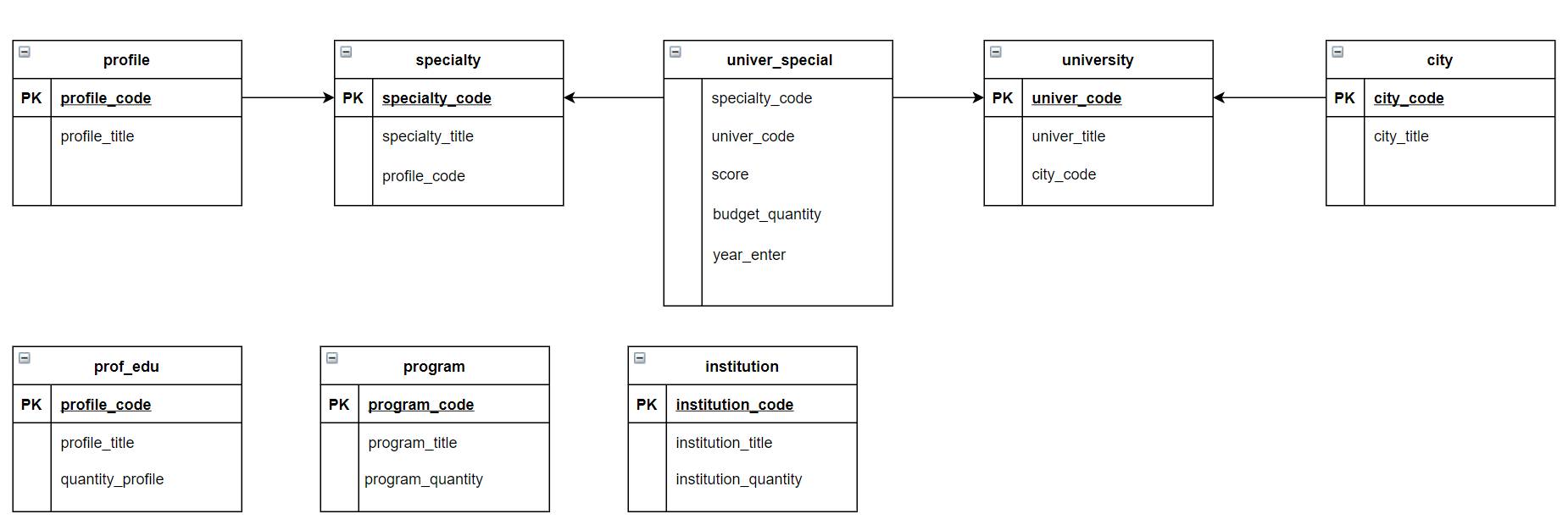


Рисунок 1.17. Нормализованная БД.

### Физическое проектирование

Для создания базы данных используется оператор CREATE DATABASE name, где «name»– имя базы данных.

CREATE TABLE city (

city\_code int primary key check(city\_code >0),

city\_title text not null);

CREATE TABLE university (

univer\_code int primary key,

univer\_title text not null,

city\_code int check(city\_code>0));

CREATE TABLE specialty (

specialty\_code text primary key,

specialty\_title text not null);

CREATE TABLE univer\_special (

univer\_code int not null,

score int not null,

budget\_quantity int check(budget\_quantity>0),

year\_enter int not null check(year\_enter>0));

ALTER TABLE university add foreign key(city\_code) REFERENCES city(city\_code);

ALTER TABLE univer\_special add foreign key(univer\_code) REFERENCES university(univer\_code);

ALTER TABLE univer\_special add foreign key(specialty\_code) REFERENCES specialty(specialty\_code);

## Сбор данных

Получение данных из сторонних сайтов проводилось с помощью документов excel.

Для примера разберем получение данных о Топ – городах России.

1. Необходимо перейти на нужную страницу и запомнить ее адрес (рисунок 1.16):

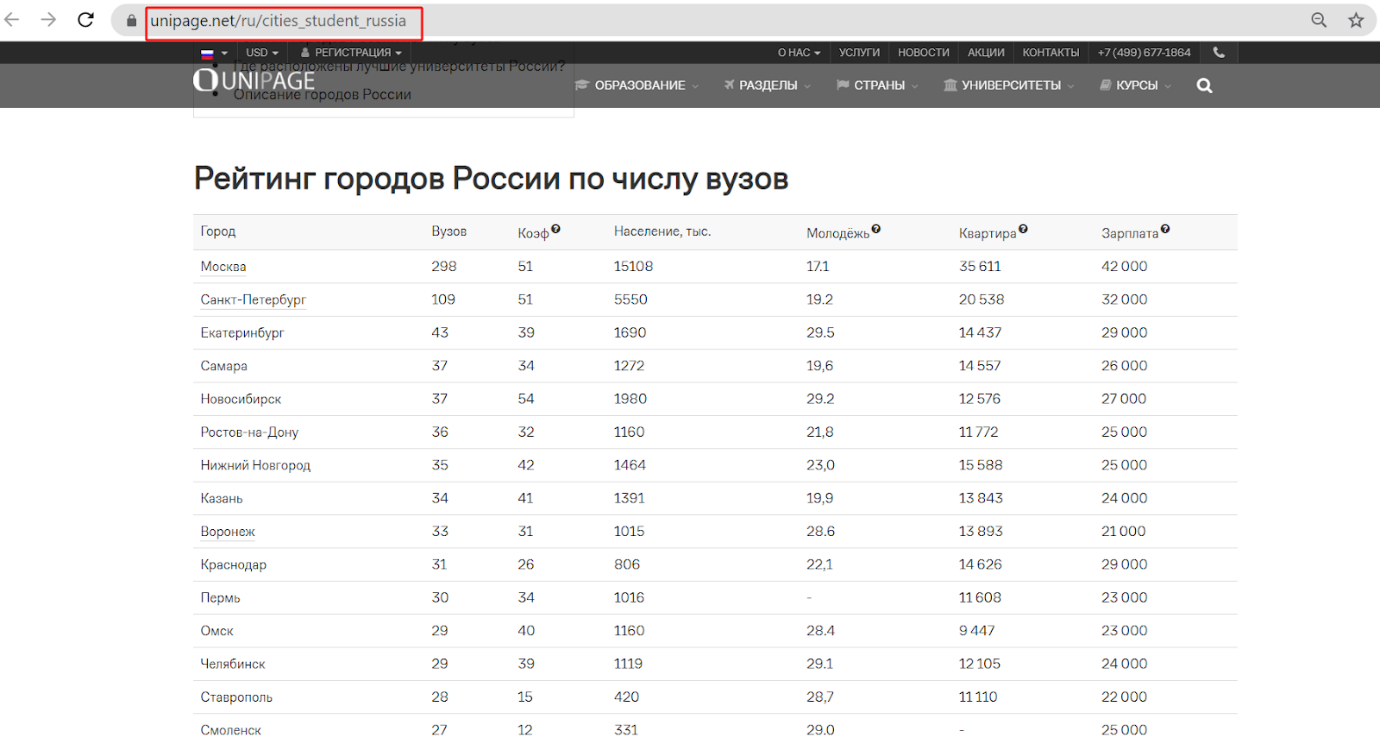


Рисунок 1.18. – Экспорт данных. Шаг 1.

1. Для того чтобы экспортировать данные с сайта в таблицу excel необходимо перейти во вкладку “Данные”, нажать на кнопку “Получение внешних данных”, выбрать пункт “Из интернета” (рисунок 1.17):

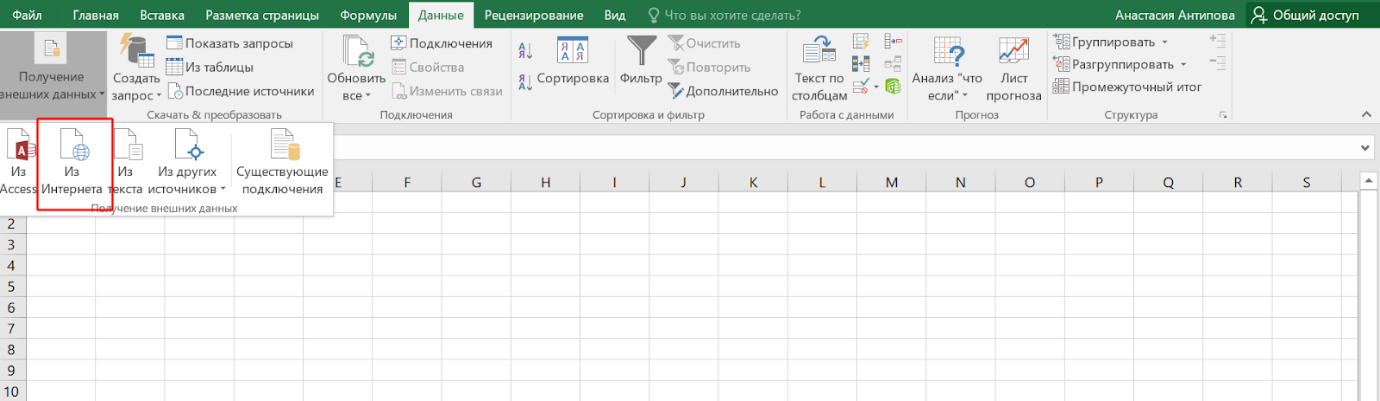


Рисунок 1.19. – Экспорт данных. Шаг 2

1. В открывшемся модальном окне указать адрес страницы, из которой необходимо вычленить информацию и нажать кнопку “Пуск” (рисунок 1.18):

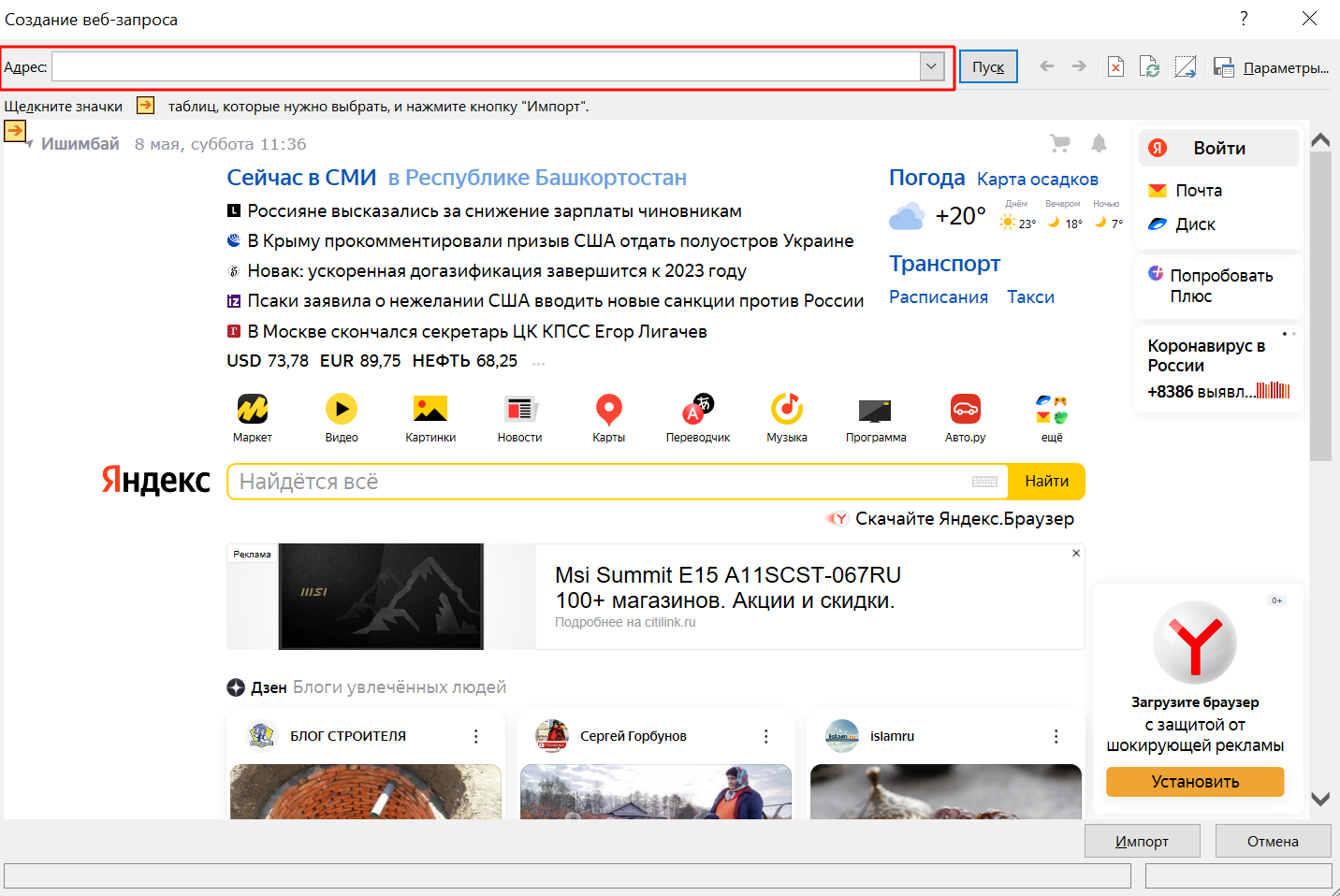


Рисунок 1.20. – Экспорт данных. Шаг 3

1. После чего откроется нужный сайт. Для того чтобы указать какая именно информация нужна, необходимо нажать на пиктограмму стрелки, после чего нажать на кнопку “Импорт” (рисунок 1.19)

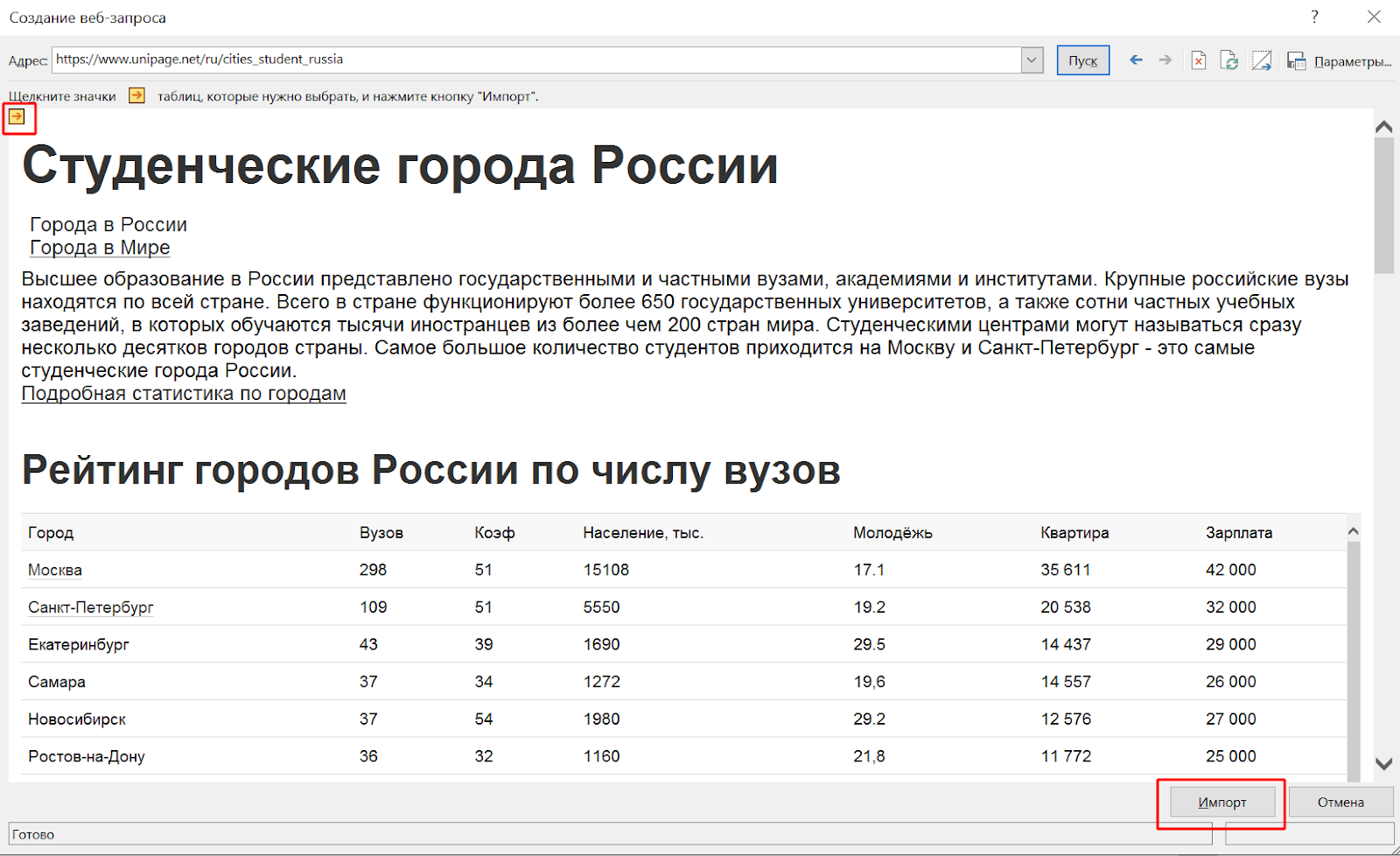


Рисунок 1.21. - Экспорт данных. Шаг 4.

1. Автоматически откроется ранее созданная таблица excel с экспортированными данными (рисунок 1.20):

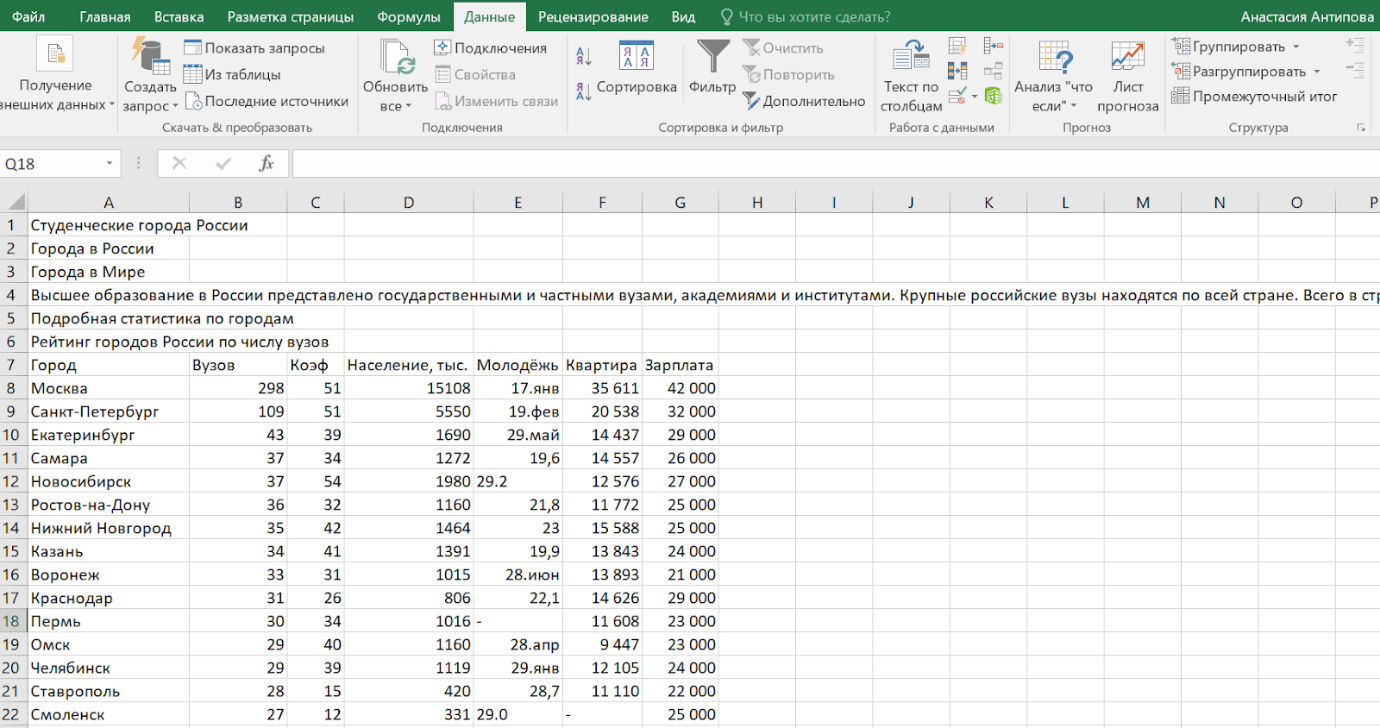


Рисунок 1.22. - Экспорт данных. Шаг 5.

1. Теперь необходимо скорректировать перенесенные данные в необходимый для дальнейшей работы вид (рисунок 1.21):

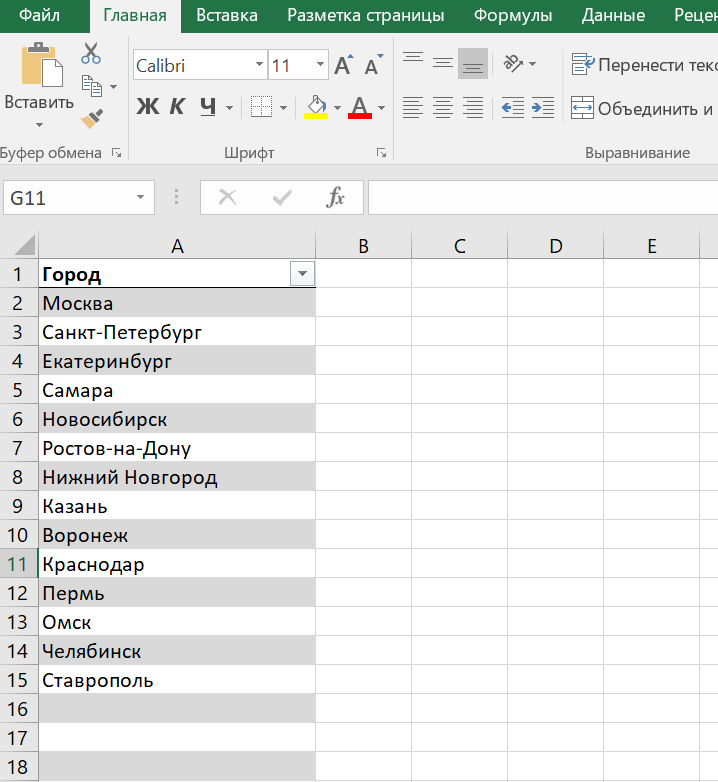


Рисунок 1.23. - Экспорт данных. Шаг 6.

Данные о проходных баллах и плане приема по каждой специальности отдельного ВУЗа прошлого года хранятся в pdf – формате (рисунок 1.22):



Рисунок 1.24. – Данные о проходном балле прошлых лет.

Для того, чтобы перенести эти данные необходимо выполнить следующий перечень действий:

1. Конвертировать документ из .pdf формата в .docx
2. Преобразовать данные в необходимый для работы вид. Были удалены заголовки таблиц и название профилей по каждой специальности, так как такие специальности имеют одинаковый код специальности.
3. Далее таблицу необходимо перенести из документа .docx формата в формат .xlsx
4. Так как одна и та же специальность (конкретнее, код специальности) может дублироваться у ВУЗа, необходимо исключить дублирующие строки, чтобы не возникало ошибок при переносе данных из .xlsx документа в базу данных (код специальности является первичным ключом в сущности БД).
5. Исключение дублирующих строк производится нажатием на кнопку “Удалить дубликаты”, находящейся во вкладке “Данные” (рисунок 1.23)

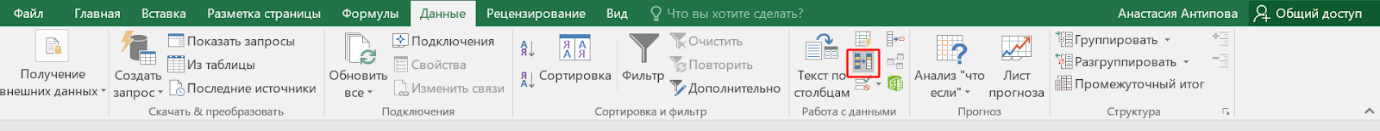


Рисунок 1.25. – Удаление дубликатов.

Данные о проходном балле каждого университета и плане приема хранится в документе .xlsx в следующем виде:

Таблица 1.4. – Вид данных о проходном балле в таблице excel.

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование специальности | Код специальности | План приема | Проходной балл |

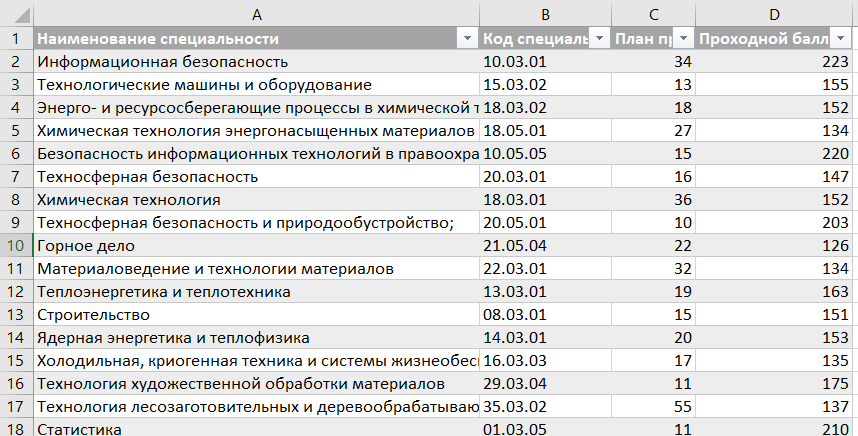


Рисунок 1.26. - Вид данных о проходном балле в таблице excel

## Импорт данных в БД

Для того чтобы импортировать данные из документа .xlsx формата в таблицу базы данных PostgreSQL необходимо сделать перечень действий:

1. Сохранить документ .xlsx формата в формате .csv
2. Полученный документ сохранить в кодировке UTF-8. Для данного действия был использован инструмент Notepad++.
3. Предоставить доступ к данному документу. Для этого необходимо перейти в раздел “Свойства” данного документа и открыть вкладку “Безопасность”. В блоке “Группы и пользователи” добавить строку “Все”, после чего нажать на “Ок” (рисунок 1.25)

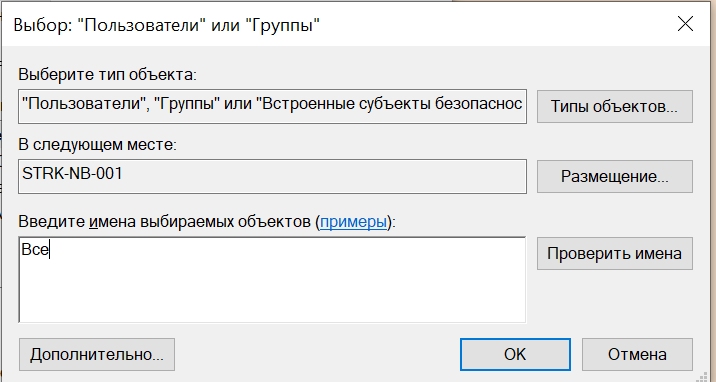


Рисунок 1.27. – Определение пользователей.

1. Предоставить полный доступ к данному документу. Нажать на кнопку “Применить”, закрыть диалоговое окно (рисунок 1.26)

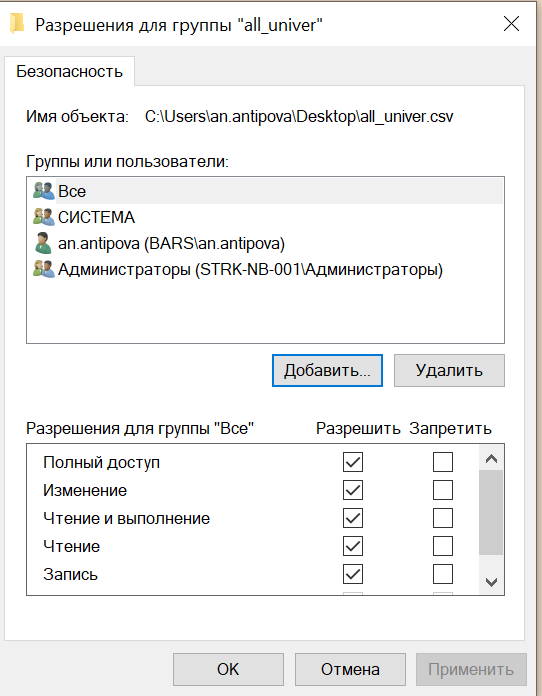


Рисунок 1.28. – Разрешение для группы

1. Перейдя в запросник pgAdmin, ввести следующий запрос:

copy university from 'C:\Users\an.antipova\Desktop\all\_univer.csv' with csv header

В данном случае импортируются данные из документа “all\_univer” формата .csv в таблицу university.

1. Провести проверку на наполненность таблицы следующим запросом:

select \* from university

Результат выполнения запроса (Рисунок 1.27):

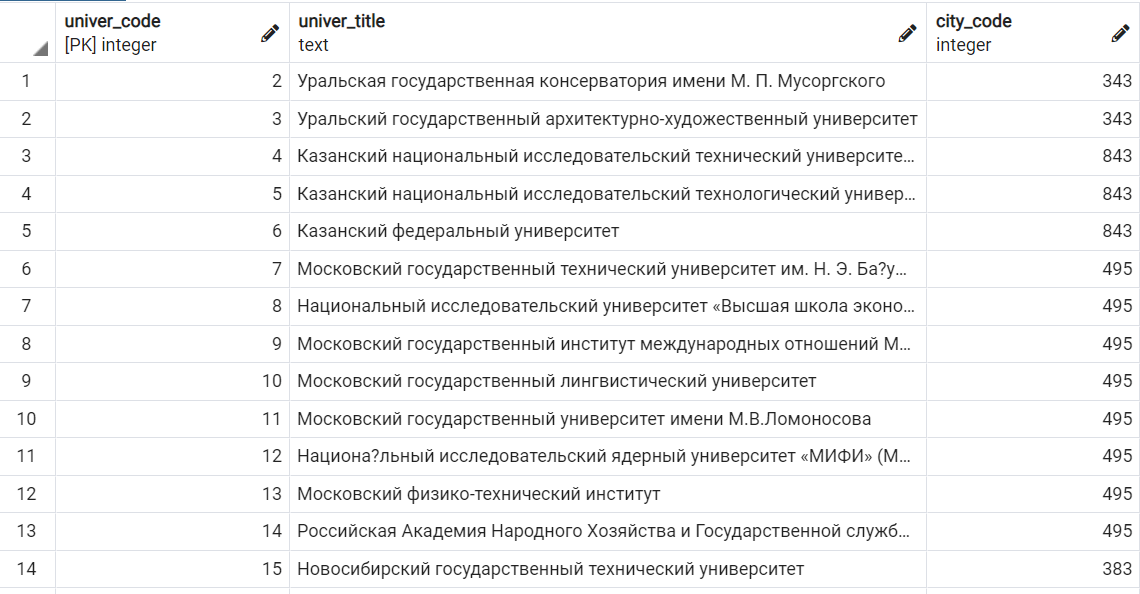


Рисунок 1.29. – Результат выполнения запроса

# РАЗРАБОТКА ПРИЛОЖЕНИЯ

Серверная часть проекта была реализована на языке JavaScript – мультипарадигменном языке программирования, поддерживающим объектно – ориентированный стиль программирования [12].

Платформой для реализации выступила интегрированная среда разработки от компании JetBrains WebStorm.

## Алгоритм определения направления обучения путем прохождения проф. ориентационного теста

Проф. ориентационный тест, который используется в разрабатываемой системе был взят из открытого источника.

Алгоритм определения направления следующий:

Для каждого направления в коде создана своя переменная. После нажатия на «Да» каждого из вопросов, та или иная переменная инкрементируется. По прохождению всего теста, система определяет переменную, в которой набралось наибольшее количество баллов, и выводит наименование профиля, за которым закреплена текущая переменная.

## Алгоритм формирования рекомендуемого списка университета

Для определения рекомендуемого списка университетов используется следующий алгоритм:

1. По каждому из университетов определяется средний проходной балл поступивших прошлого года;
2. Далее это значение сравнивается со значением той переменной, которую ввел пользователь;
3. После разница этих переменных ранжируется в порядке возрастания
4. По итогу выводятся наименования университетов в том порядке, который получились в п.3.

## Открытие проекта

Для того чтобы запустить проект необходимо выполнить перечень действий:

* + - * 1. Открыть Terminal и в командной строке указать: npm run server (рисунок 2.1)

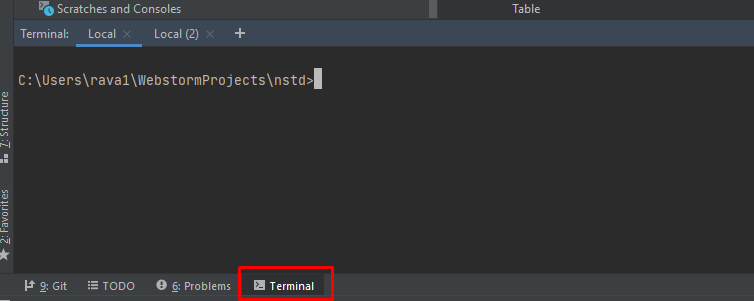


Рисунок 2.1. – Открытие проекта. Шаг 1.

* + - * 1. Далее необходимо создать доп. вкладку Local2 и ввести «cd client», после загрузки указать «npm start» (рисунок 2.2)

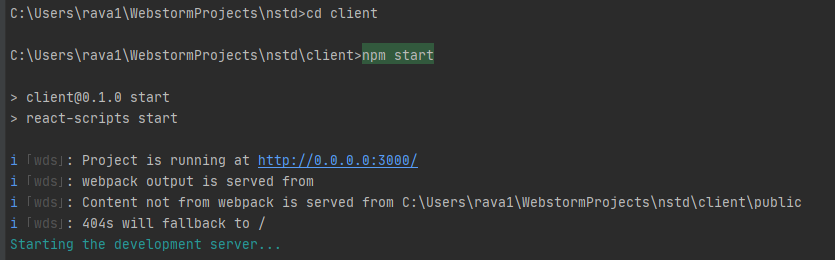


Рисунок 2.2. - Открытие проекта. Шаг 2.

* + - * 1. После выполненных действий откроется браузер с разработанным сайтом

## Обзор проекта

При реализации данного проекта было создано несколько папок:

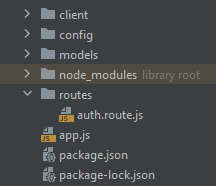


Рисунок 2.3. – Обзор папок проекта

1. Clien – весь фрон;

2. Config – конфиг;

3. Models – модели;

4. node\_modules – библиотеки;

5. routes – API – запросы;

6. app.js – основной файл;

7. package.json – все зависимости между библиотеками.

### Routes

Папка, в которой расположены файлы, внутри которых описаны пути API – запросов и обработка данных. Именно через эти запросы фронт общается с бэком (получает данные, отправляет их и т.д.)

Для начала необходимо импортировать все необходимые пакеты:

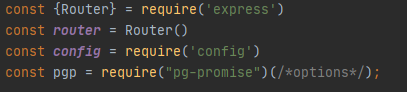


Рисунок 2.4. – Импорт пакетов

Далее создать метод:

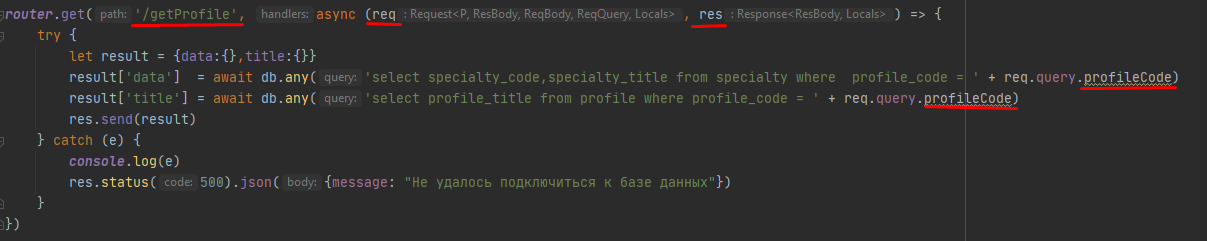


Рисунок 2.5. – Создание метода

Где:

/getProfile – ссылка по которой будет отрабатывать этот метод (полный путь будет localhost:5000/api/getProfile , /api определяется в файле app.js )

Рисунок 2.6. – ссылка API - запроса

req (request) – то, что получает сервер;

res (response) – то, что сервер возвращает;

req.query.profileCode – параметр, который необходимо указать в ссылке через знак вопроса



Рисунок 2.7. – Пример ссылки

Внутри создается переменная с типом object, и при попытке вывода данных отобразится результат:

{

data:{},

title:{}

}

После создается два запроса к БД

Перед каждым из них стоит await, это необходимо для того чтобы JS подождал пока БД вернет ответ, а не пошел дальше по коду и не вернул пустую переменную.

Далее записываются данные в переменную и возвращаются.

Ответ приходит в виде:

{

    "data": [

        {

            "specialty\_code": "01.03.01",

            "specialty\_title": "Математика "

        },

        {

            "specialty\_code": "01.03.02",

            "specialty\_title": "Прикладная математика и информатика"

        },

        {

            "specialty\_code": "01.03.03",

            "specialty\_title": "Механика и математическое моделирование"

        }

    ],

    "title": [

        {

            "profile\_title": "Математические и естественные науки"

        }

    ]

}

Фронт делает запрос, получает данные и помещает их в таблицу.

### Routes.js

Здесь происходит вся маршрутизация

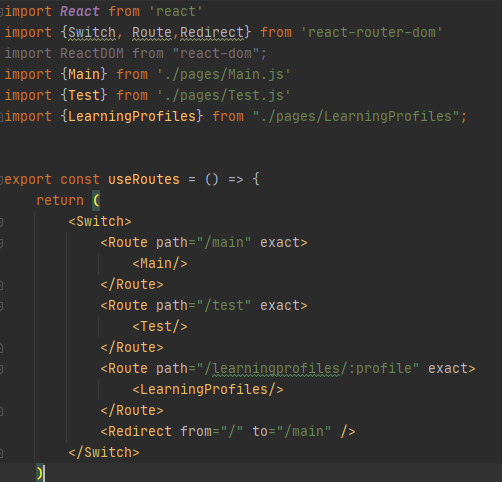


Рисунок 2.8. – Маршрутизация.

Вначале импортируются страницы, и внутри указываются функции, что если ссылка = localhost:3000/main, то открывается страницы Main и т.д.

/learningprofiles/:profile

:profile –переменная;



Рисунок 2.9. – Пример ссылки.

Данная переменная используется для того, чтобы далее на страничке сделать запрос к БД.

### Main

Импортируется hook useRef:



Рисунок 2.10. –Импорт useRef:

Создается следующий метод:



Далее используется стрелочная функция [13], по функционалу схожа на следующее:

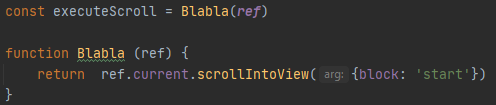


Рисунок 2.11. –Стрелочная функция.

Различие в том, что переменную объявить можно прямо в константе:



Рисунок 2.12. – Объявление переменной.

Объявление констант:

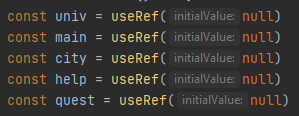


Рисунок 2.13. –Объявление констант.

Далее создается ref на нужных элементах (куда необходимо перейти по нажатию):



Рисунок 2.14. – Создание ref.

Объявить обработку нажатия на элементы в оглавлении:



Рисунок 2.15. – Обработка нажатия кнопки.

Те же самые действия необходимо выполнить со стрелочной функцией, так как внутри можно писать только ссылку на функцию, а не вызывать ее саму (то есть вызов функции этот как executeScroll(city), а ссылка на функцию — это просто executeScroll, но в данном случае она не подходит, так как по задаче необходимо передать в нее аргумент).

Так же в css необходимо создать следующий элемент, чтобы переход был плавным:

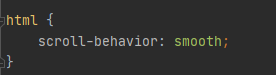


Рисунок 2.16. – Прокрутка.

А также создать элементы:



Рисунок 2.17. – Создание элементов.

Далее представлена обработка нажатия кнопки – это функция, которая открывает другую страницу:



Рисунок 2.18. – Обработка нажатия кнопки.

Так как никаких аргументов передавать не нужно, указывается Test:

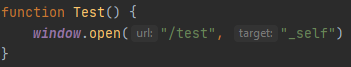


Рисунок 2.19. – Объявление функции Test.

### Test

Чтоб не утяжелять код создаем элемент, в котором есть и вопрос, и кнопки:



Рисунок 2.20. – Объявление функции Question.

Так как в проф. ориентационном тесте 30 вопросов и в каждом разный текст, и разный профиль, то необходимо использовать переменные. Поэтому внутри h1 указывается props.question, а при использовании элемента question меняется.



Рисунок 2.21. – Вызов функции Question.

В name указываются профили, а обработка нажатия — это функция для подсчета баллов по каждому профилю (для подсчета баллов используется стандартный switch-case).

### LearningProfiles

В данной расположены запросы к БД.

Создается кастомная таблица (указывается класс, потому что необходимо переопределить несколько методов и еще добавить собственные):



Рисунок 2.22. – Создание кастомной таблицы.

Конструктор:

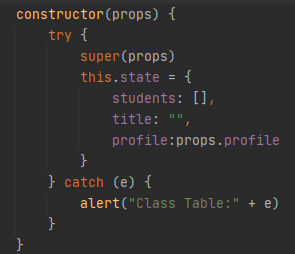


Рисунок 2.23. – Объявление конструктора.

Вышеописанный конструктор срабатывает после того, как элемент уже отрисовался. Далее создается запрос к серверу, который в свою очередь делает запрос к БД.

Указывать данный запрос нужно именно в этот метод так как запросы – асинхронная вещь, то есть пока JS ждет ответа от сервера, он идет выполнять весь остальной код, и нужно использовать await, но, чтоб использовать await нужно чтобы функция была async.

Рисунок 2.24. – Запрос к серверу.

В вышерасположенном коде используется this.state.profile, который указывался в конструкторе, но там он также подтягивается из props.

В html указано:



Рисунок 2.25. – Вызов profile.

И тут значение берется из переменной:



Рисунок 2.26. – Вызов useParams.

В вышерасположенном рисунке используется useParams() – это hook, который берет параметры из ссылки и используется его в routes(клиентская часть):

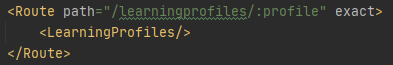


Рисунок 2.27. – Использование useParams в routes.

Далее описывается очень важный момент: из ссылки берется параметр



Рисунок 2.28. – Описание ссылки

В данном случае 1, затем эта единичка уходит в элемент и сохраняется в this.state:



Рисунок 2.29. – Элемент profile.

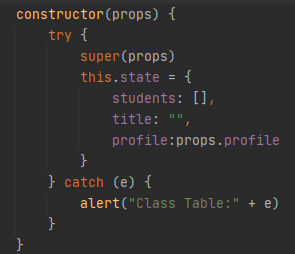


Рисунок 2.30. – Сохранение элемента в this.state

Далее этот используется для запроса:



Рисунок 2.30. – Описание запроса к серверу

Затем в API эта переменная вызывается в запросе к БД:

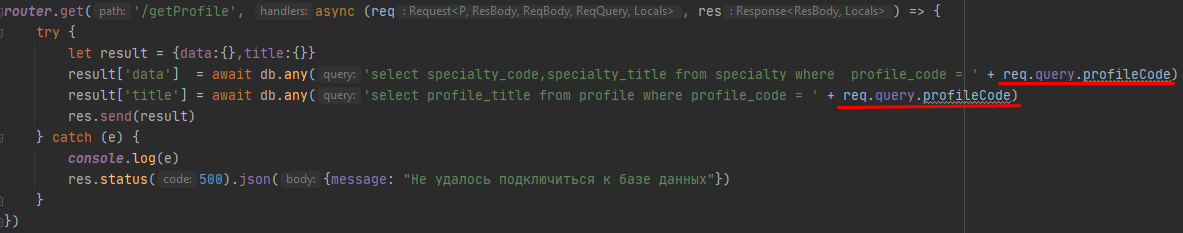


Рисунок 2.30. – Запрос к БД.

И получается sql запрос вида:

select specialty\_code, specialty\_title from specialty where profile\_code = 1

А при таком запросе БД нам выдает специальности по заданному профилю:

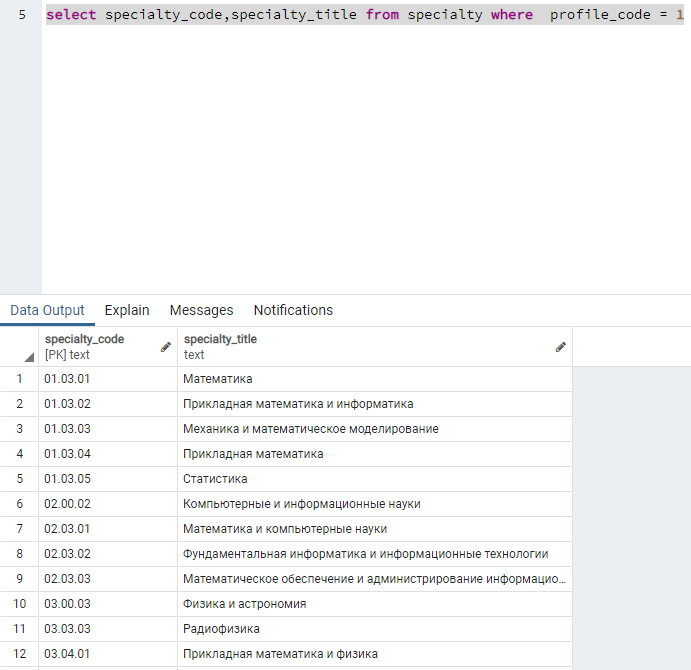


Рисунок 2.30. – Результат выполнения запроса в БД.

Потом данная таблица возвращается в элемент и превращается в таблицу вида, который представлен ниже:

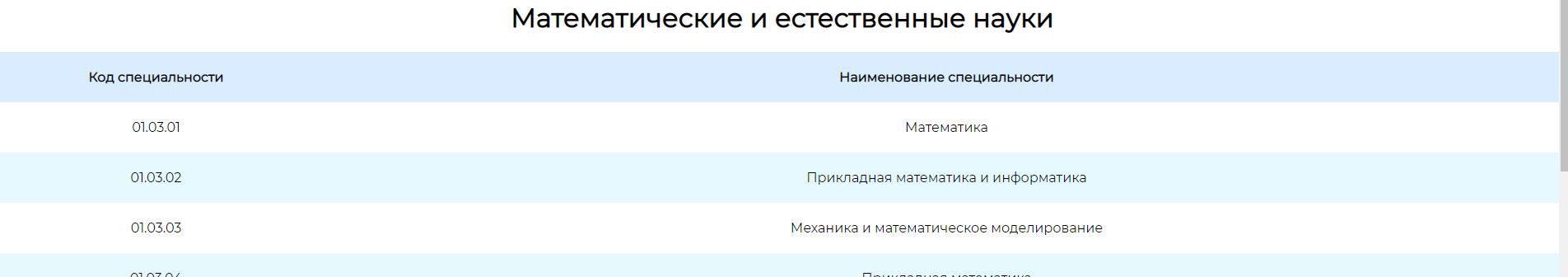


Рисунок 2.30. – Результат запроса на разрабатываемом портале.

Метод render отрисовывает все на страничке, в ниже представленном рисунке описывается и заголовок, и данные:



Рисунок 2.30. – Метод render.

# ЗАКЛЮЧЕНИЕ

По результату выполнения выпускной квалификационной работы была разработана рекомендательная система подбора университетов для абитуриентов, а также была проведена проверка и тестирование данной рабочей среды.

В результате разработки рекомендательной системы для пользователей была реализована возможность бесплатного прохождения проф. ориентационного теста, с возможностью просмотра рекомендательного направления обучения для поступления, а также предусмотрен функционал по подбору высшего учебного заведения по заданным баллам ЕГЭ.

Несмотря на то, что рекомендательная система подбора университетов для абитуриентов является учебной, она полностью готова к эксплуатации, в то же время она может быть доработана при необходимости.

# СПИСОК ЛИТЕРАТУРЫ

1. Теоретические аспекты профориентации подростков [Электронный ресурс]. – Режим доступа : https://sch17uz.mskobr.ru/. - (Дата обращения: 03.06.2021).
2. Справочник профессий [Электронный ресурс] / Министерство труда и социальной защиты Российской Федерации. – Режим доступа : http://spravochnik.rosmintrud.ru/. – (Дата обращения: 14.05.2021).
3. Карлберг, К. Бизнес-анализ с использованием Excel / К. Карлберг. - М.: Вильямс, 2016. - 576 c.
4. Поступи. Онлайн [Электронный ресурс]. – Режим доступа : https://postupi.online/. - (Дата обращения: 09.03.2021).
5. Вузотека [Электронный ресурс]. – Режим доступа : https://vuzoteka.ru/ (Дата обращения: 09.03.2021).
6. Edunetwork Онлайн [Электронный ресурс]. – Режим доступа : https://vuz.edunetwork.ru/ (Дата обращения: 09.03.2021).
7. Генератор [Электронный ресурс]. – Режим доступа : https://sales-generator.ru/ (Дата обращения: 16.04.2021).
8. Федоров И.Г. Моделирование бизнес-процессов в нотации BPMN 2.0. М.: МЭСИ, 2013. 264 с
9. Дунаев, В. В. Базы данных. Язык SQL для студента / В.В. Дунаев. - М.: БХВ-Петербург, 2017. - 288 c.
10. Туманов, В.Е. Основы проектирования реляционных баз данных / В.Е. Туманов. - М.: Бином, 2018. - 420 c
11. Уорден, К. Новые интеллектуальные материалы и конструкции. Свойства и применение / К. Уорден. - М.: Техносфера, 2018. - 224 c.
12. Бибо Бер , Кац Иегуда jQuery. Подробное руководство по продвинутому JavaScript; Символ-плюс - М., 2017. - 624 c.
13. Дронов Владимир JavaScript. Народные советы; БХВ-Петербург - М., 2016. - 458 c.