МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Информационных систем и технологий

Специальность 1-40 01 01 «Программное обеспечение информационных

технологий»

Специализация 1-40 01 01 10 «Программное обеспечение информационных

технологий (программирование интернет-приложений)»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовому проекту на тему:**

Web-приложение «Изучение английского языка»

Выполнил студент Белашков Василий Александрович

(Ф.И.О.)

Руководитель проекта ст. преп. Дубовик М. В.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Заведующий кафедрой к.т.н., доц. Смелов В.В.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Консультанты ст. преп. Дубовик М. В.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Норм контролёр ст. преп. Дубовик М. В.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Минск 2023

Содержание

[Введение 3](#_Toc104198214)

[1. Постановка задачи 4](#_Toc104198215)

[1.1 Аналитический обзор литературы 4](#_Toc104198216)

[1.2 Анализ прототипов 6](#_Toc104198217)

[1.3 Функциональные требования 7](#_Toc104198218)

[2. Проектирование приложения 9](#_Toc104198219)

[2.1 Проектирование базы данных 9](#_Toc104198220)

[2.2 Идентификация и авторизация 13](#_Toc104198221)

[3. Разработка приложения 14](#_Toc104198222)

[3.1 Реализация базы данных 14](#_Toc104198223)

[3.2 Реализация серверной части. 14](#_Toc104198224)

[3.3 Веб-сокеты серверная и клиентская часть 17](#_Toc104198225)

[3.4 Проектирование клиентской части приложения 18](#_Toc104198226)

[3.5 Конечные точки 19](#_Toc104198227)

[4. Тестирование 20](#_Toc104198228)

[4.1 Страница поиска университета 20](#_Toc104198229)

[4.2 Форма входа сайта университета 21](#_Toc104198230)

[4.3 Форма регистрации 22](#_Toc104198231)

[4.4 Форма заполнения личной информации 24](#_Toc104198232)

[5. Руководство пользователя 26](#_Toc104198233)

[5.1 Сайт для подбора университета 26](#_Toc104198234)

[5.2 Авторизация и регистрация 27](#_Toc104198235)

[5.3 Сайт университета для пользователя 28](#_Toc104198236)

[5.4 Сайт университета для члена приёмной комиссии 32](#_Toc104198237)

[6. Заключение 35](#_Toc104198238)

[7. Список использованной литературы 36](#_Toc104198239)

[Приложение А 37](#_Toc104198240)

[Приложение Б 40](#_Toc104198241)

# Введение

В современном мире знание английского языка является важным условием для успешной карьеры, обучения за границей, путешествий и общения с иностранными друзьями и коллегами. Однако, не всегда у нас есть возможность посещать курсы и занятия по изучению английского языка в оффлайн формате. В таком случае, web-приложения для изучения английского языка становятся наиболее удобным и доступным решением.

Целью данного курсового проекта является разработка web-приложения для изучения английского языка. Оно будет позволять пользователям улучшать свои навыки и знания в любое удобное время и в любом месте. В рамках проекта будет рассмотрено несколько аспектов, необходимых для создания и успешного функционирования web-приложения.

Первым этапом будет разработка концепции приложения и его функциональных требований. Здесь мы определим основные цели и задачи, которые должно решать приложение, а также его целевую аудиторию. Кроме того, необходимо обозначить функции и возможности, которые будут включены в приложение.

Вторым этапом является проектирование приложения, включающее в себя разработку структуры базы данных, взаимодействия пользователя с интерфейсом приложения и создание системы обработки и хранения данных. Также здесь определяется дизайн приложения и разработка интерфейсов, которые будут максимально удобны и понятны для пользователей.

Третий этап – программирование и тестирование приложения. Здесь будет создан код, который обеспечит функциональность и работу приложения, а также протестирован на соответствие функциональным требованиям.

Кроме того, приложение будет содержать возможности для ведения статистики прогресса пользователя. Это позволит пользователю отслеживать свой прогресс и узнавать о своих достижениях.

Таким образом, данное web-приложение будет полезным и удобным инструментом для изучения английского языка, доступным в любое время и в любом месте. Разработка и успешное функционирование приложения потребует учета многих аспектов и внимательной работы над каждым этапом процесса разработки. Однако, благодаря правильно выбранной концепции, дизайну и функциональности, приложение сможет помочь пользователям в достижении своих целей в области изучения английского языка.

# Постановка задачи

По заданию курсового проекта необходимо разработать web-приложение, позволяющее администратору и пользователю авторизоваться; администратору добавлять, изменять и удалять словари, слова в словарях, уроки и задания в уроках; пользователю добавлять, изменять и удалять словари и слова в них, изучать слова в словарях администратора, изучать уроки и задания добавленные администратором, изучать слова в соревновательном режиме, а именно в виде квиза.

## 1.1 Аналитический обзор литературы

Одним из ключевых моментов в разработке программного обеспечения является изучение аналогов, выявление достоинств и недостатков в них. С помощью анализа аналогов можно выделить функционал, который обязательно должен присутствовать в web-приложении. Это необходимо для построения каркаса будущего приложения. Также анализ помогает выделить недостатки и избежать их в собственной реализации.

DUOLINGO.COM

Данное web-приложение имеет следующие возможности:

* изучение грамматических правил и слов по определенным темам в виде игрового процесса;
* тематические диалоги, которые позволяют приблизиться к жизненным ситуациям;
* личная статистика прогресса и статистика друзей;
* система достижений и бонусов за получение этих достижений.

Интерфейс web-приложения представлен на рисунке 1.1.

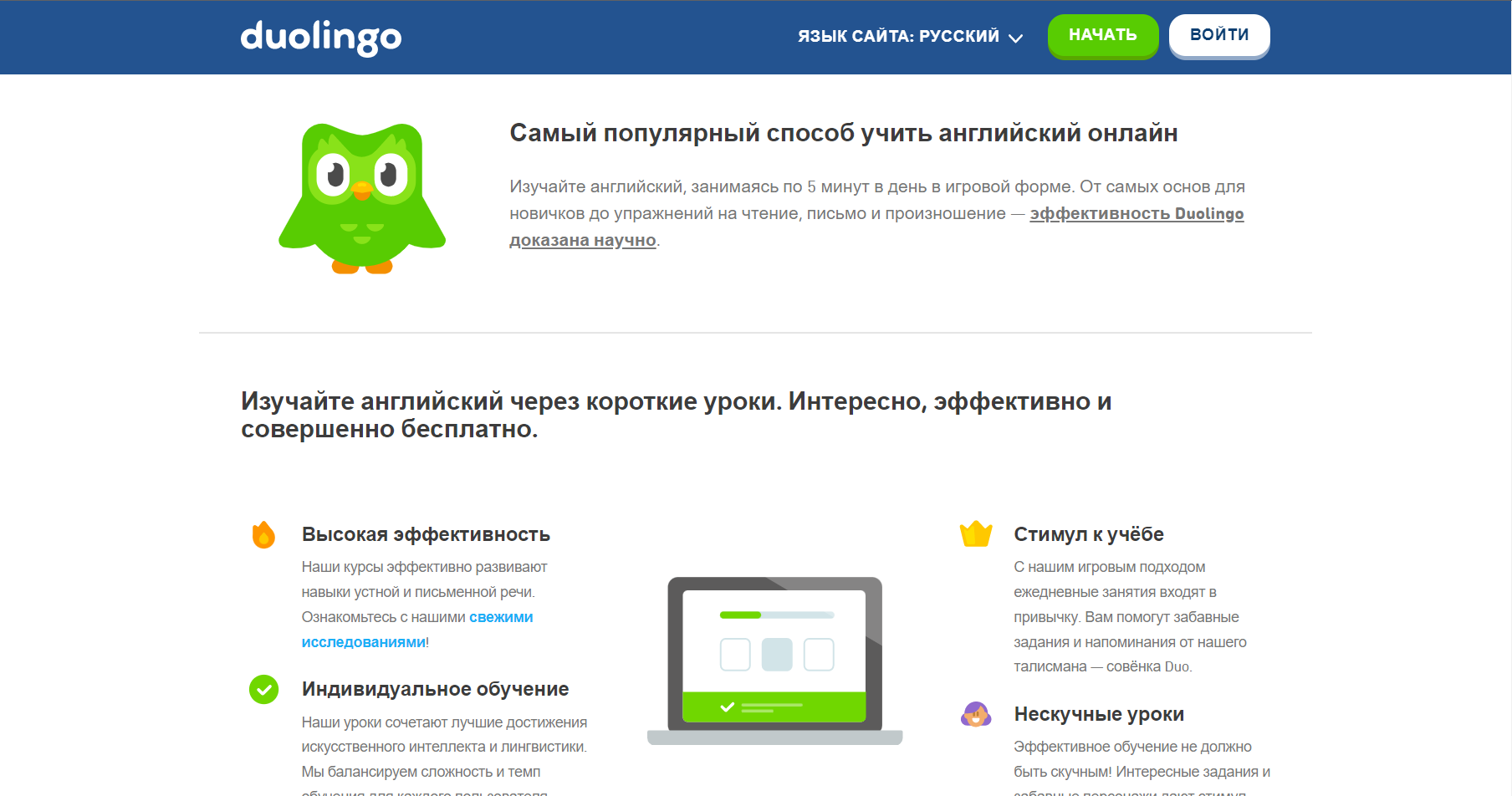


Рисунок 1.1 – Web-приложение duolingo.com

Из недостатков необходимо отметить то, что имеющийся контент позволяет получить только начальные навыки овладения языком. Также невозможно настроить приложение под свои навыки. У приложения есть и плюсы, например, за различные достижения пользователю вручаются награды (за количество выученных слов, за ежедневные посещения). Отслеживая статистику других пользователей, повышается мотивация изучения и прохождения уровней дальше.

POLIGLOT16.RU

Приложение имеет следующие возможности:

* практические и тестовые задания по грамматике;
* изучение слов по тематическим словарям;
* добавление собственных словарей;
* настройка режимов обучения.

Интерфейс web-приложения представлен на рисунке 1.2.

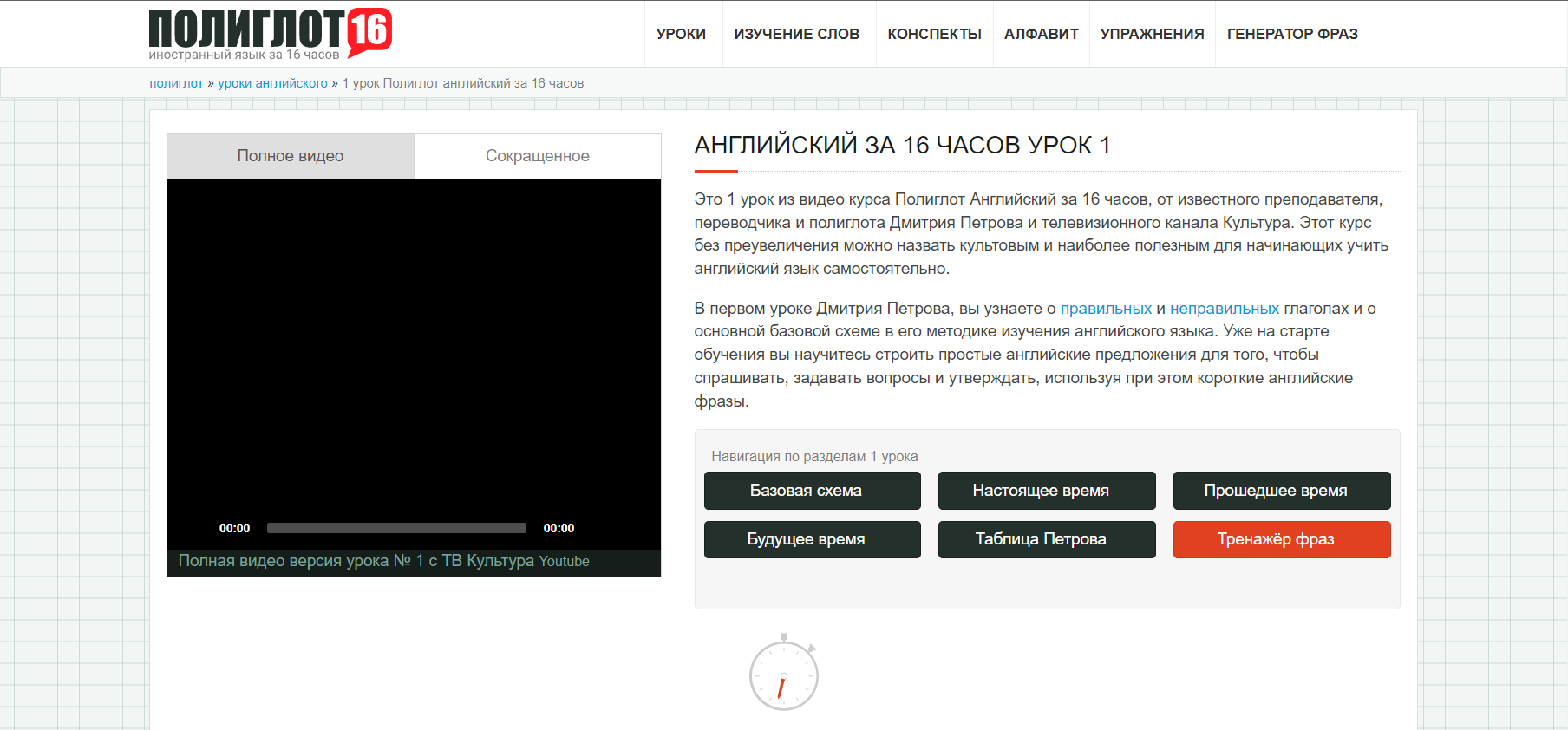


Рисунок 1.2 – Web-приложение poliglot16.ru

Довольно устаревший дизайн и не обновляемая учебная база, что делает web-приложение быстро выходящим из актуальности.

PUZZLE-ENGLISH.COM

Данное приложение так же было разработано под мобильные операционные системы IOS и Android. Имеет следующие возможности:

* уроки по грамматическим правилам и расширение словарного запаса;
* конспект с грамматическими правилами;
* тематические истории в процессе прохождения, которых хорошо запоминаются слова за счёт образования ассоциаций;
* видео-уроки с текстовым сопровождением;
* возможность установить напоминание о занятии на любое время.

Интерфейс web-приложения представлен на рисунке 1.3.

К недостаткам данного web-приложения можно отнести:

* основная часть контента является платной;
* невозможность изучать грамматику и лексику отдельно;
* довольно короткие уроки, за которые сложно усвоить тему.

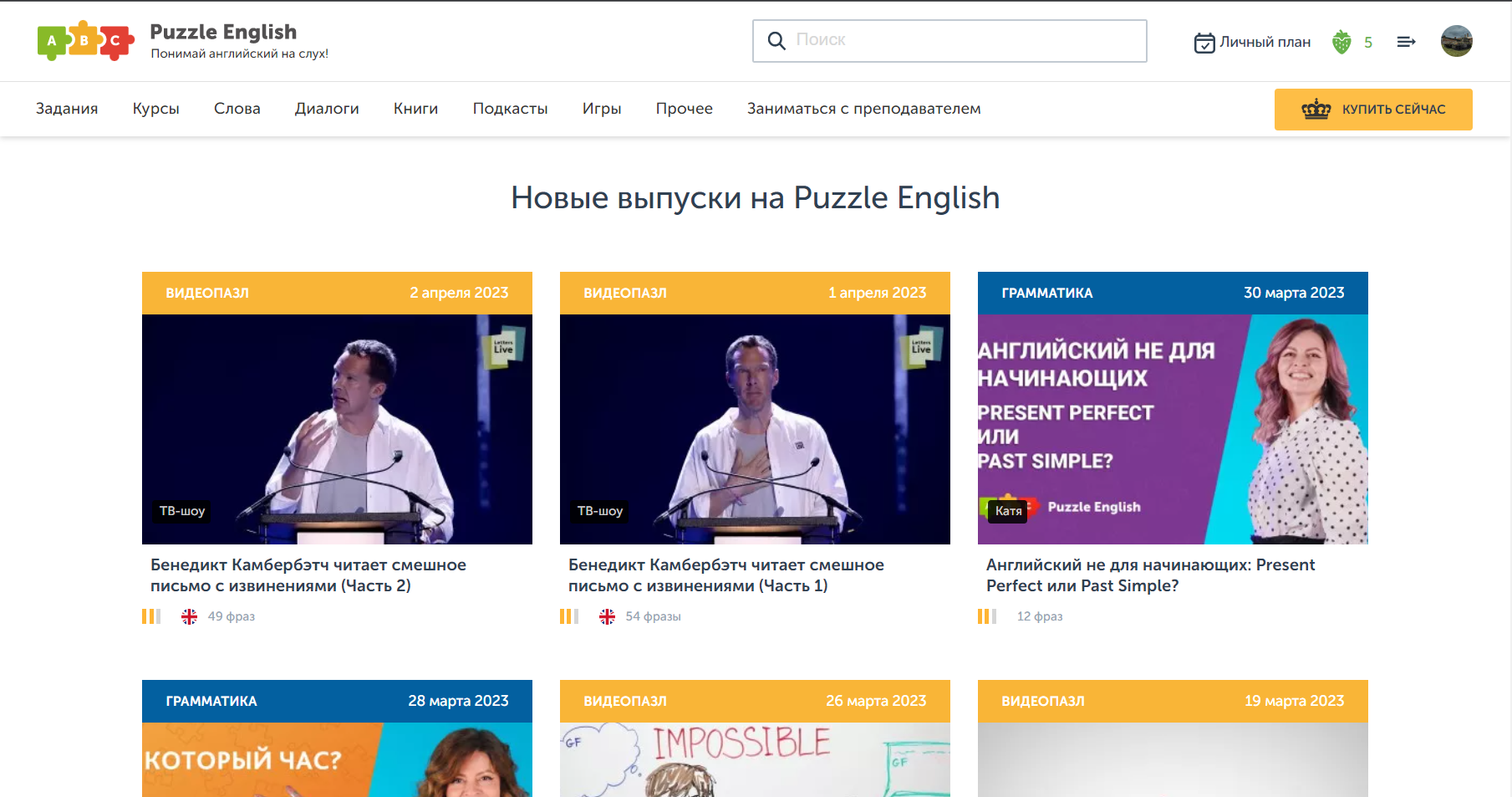


Рисунок 1.3 – Web-приложение puzzle-english.com

## 1.2 Анализ прототипов

Схожесть рассмотренных программных средств заключается в том, что все они имеют как грамматические задания, так и задания на расширение лексикона. Также эти приложения содержат контент, который позволяет получить лишь начальные знания языка.

Главные отличия программ кроются в их функциональности. Так, например, в приложениях Duolingo и Puzzle-English изучение грамматики и словарного запаса проходит в модульной форме, в отличии от приложения Poliglot16. Только в приложении Duolingo имеется статистика прогресса, в приложении Poliglot16 возможность добавлять собственные словари, а в приложении Puzzle-English есть грамматические конспекты. Также приложение Puzzle-English основную часть своего контента предлагает на платной основе.

Таким образом, можно сделать вывод о том, что в каждом приложении есть свои плюсы и минусы. Но, разумеется, каждому разработчику хочется, чтобы его приложение не имело отрицательных качеств, так как это сильно влияет рейтинг приложения, от чего напрямую зависит прибыль и статус самого разработчика.

Анализируя ранее приведённые примеры, можно составить основные функциональные особенности для подобного типа приложений:

* регистрация и авторизация пользователя;
* тренировка грамматики и расширение словарного запаса;
* база грамматических заданий и словарей;
* настройки обучения;
* статистика пользователя.

Как сказано выше, программа должна быть предназначена для различной аудитории пользователей. Это значит, что приложение должно быть простое и иметь доступный дизайн.

Все эти пункты и были учтены при выполнении данного курсового проектирования.

## 1.3 Функциональные требования

Приложение должно иметь разделение ролей. Для каждой роли были разработаны UML-диаграммы вариантов использования. При открытии сайта пользователь не авторизован. Диаграмма вариантов использование неавторизованного пользователя отображена на рисунке 1.5.

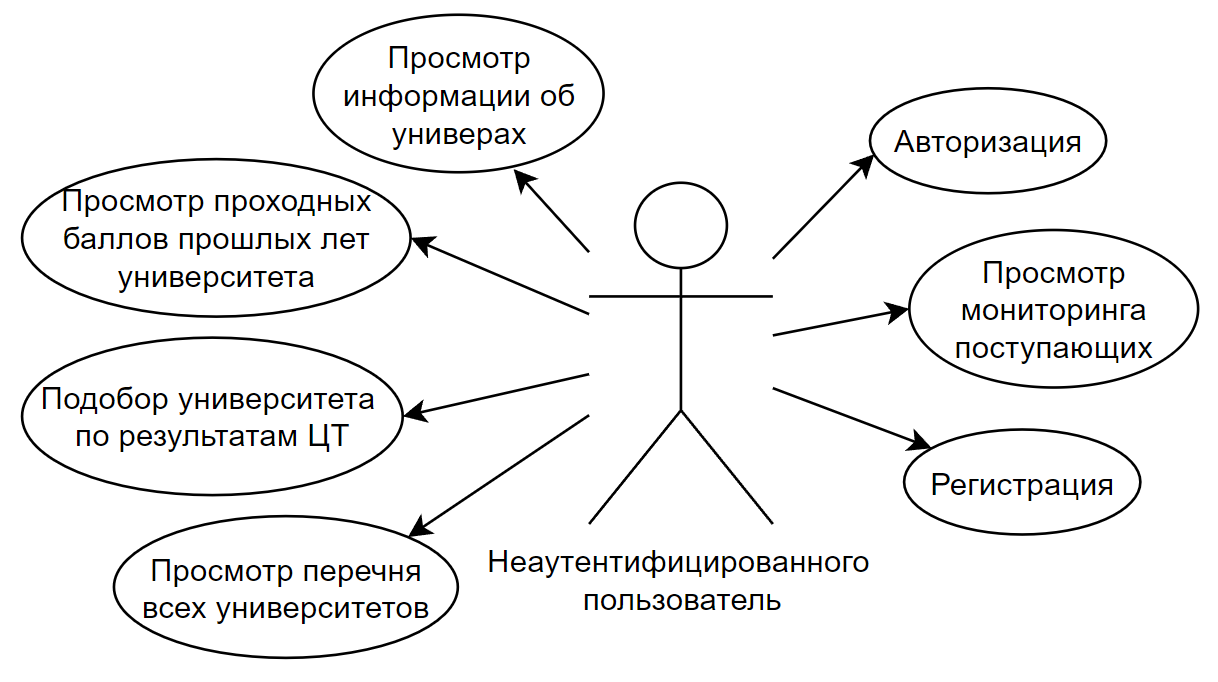


Рисунок 1.5 – Диаграмма использования неавторизованного пользователя

Как видно из диаграммы, неавторизованного пользователь может просматривать перечень всех университетов, а также подобрать те, вероятность в которые поступить выше, просматривать их пороги предыдущих лет. На части сайта, которая отвечает за конкретный универ, а именно БГТУ, неавторизованный пользователь может просматривать мониторинг поступающих (бюджет/платное), а для дальнейших действий необходимо зарегистрироваться, либо авторизоваться. При попытке подать заявку на поступление, пользователя будет перенаправлять на страницу регистрации и авторизации. При попытке зайти на страницу, требующую авторизации (например, путём явного ввода url), так же будет переводиться на страницу авторизации.

Роль каждой учётной записи определена в базе данных. Создать администратора на сайте невозможно. Итак, зарегистрировав и войдя в аккаунт, пользователь получает роль авторизованного пользователя. Диаграмма вариантов использования для этой роли отображена на рисунке 1.6.

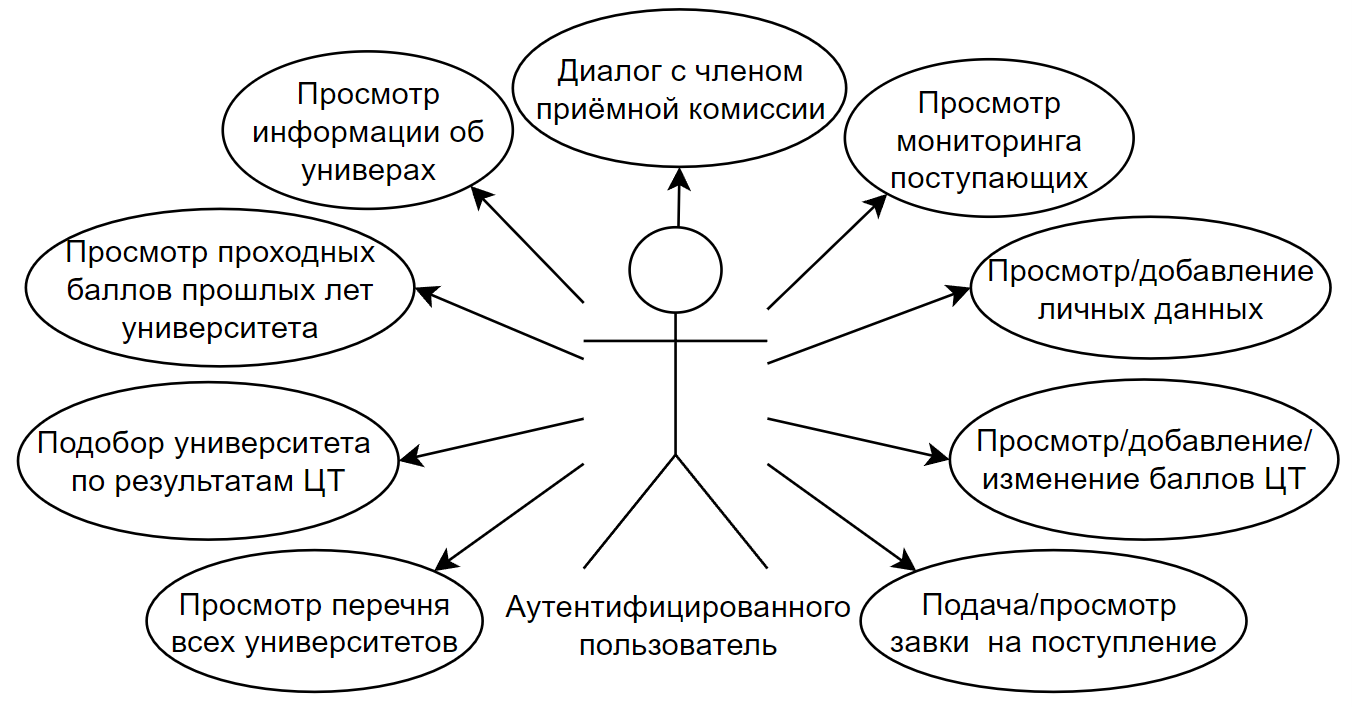


Рисунок 1.6 – Диаграмма использования авторизованного пользователя

В приложении есть роль администратора или же члена приёмной комиссии. Для этой роли диаграмма вариантов использования отображена на рисунке 1.7.

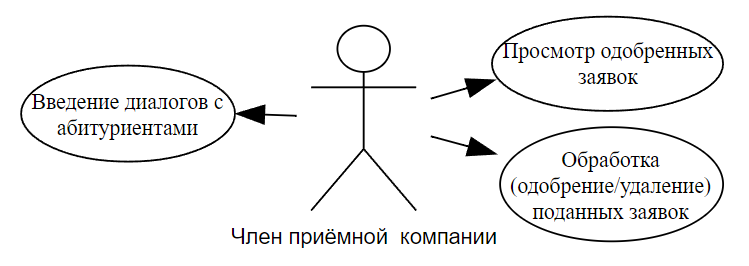


Рисунок 1.7 – Диаграмма использования для работника университета

На основе данных UML‑диаграмм и словесного описания функциональных требований строится вся бизнес логика программы.

# Проектирование приложения

Для проектирования приложения используется платформа NodeJS, для хранения данных выбрана реляционная база данных MS SQL. Для взаимодействия используется ORM, такой подход уменьшит количество работы при смене базы данных.

## 2.1 Проектирование базы данных

При проектировании базы данных разработана таблицы для хранения данных справочной информации ВУЗов пользователя.

Приложение имеет таблицы: university\_data, faculty\_data, speciality\_data, entry\_threshold, authorization\_data, users\_data, users\_marks, overall\_rating.

Диаграмма таблиц базы данных отображена на рисунке 2.1

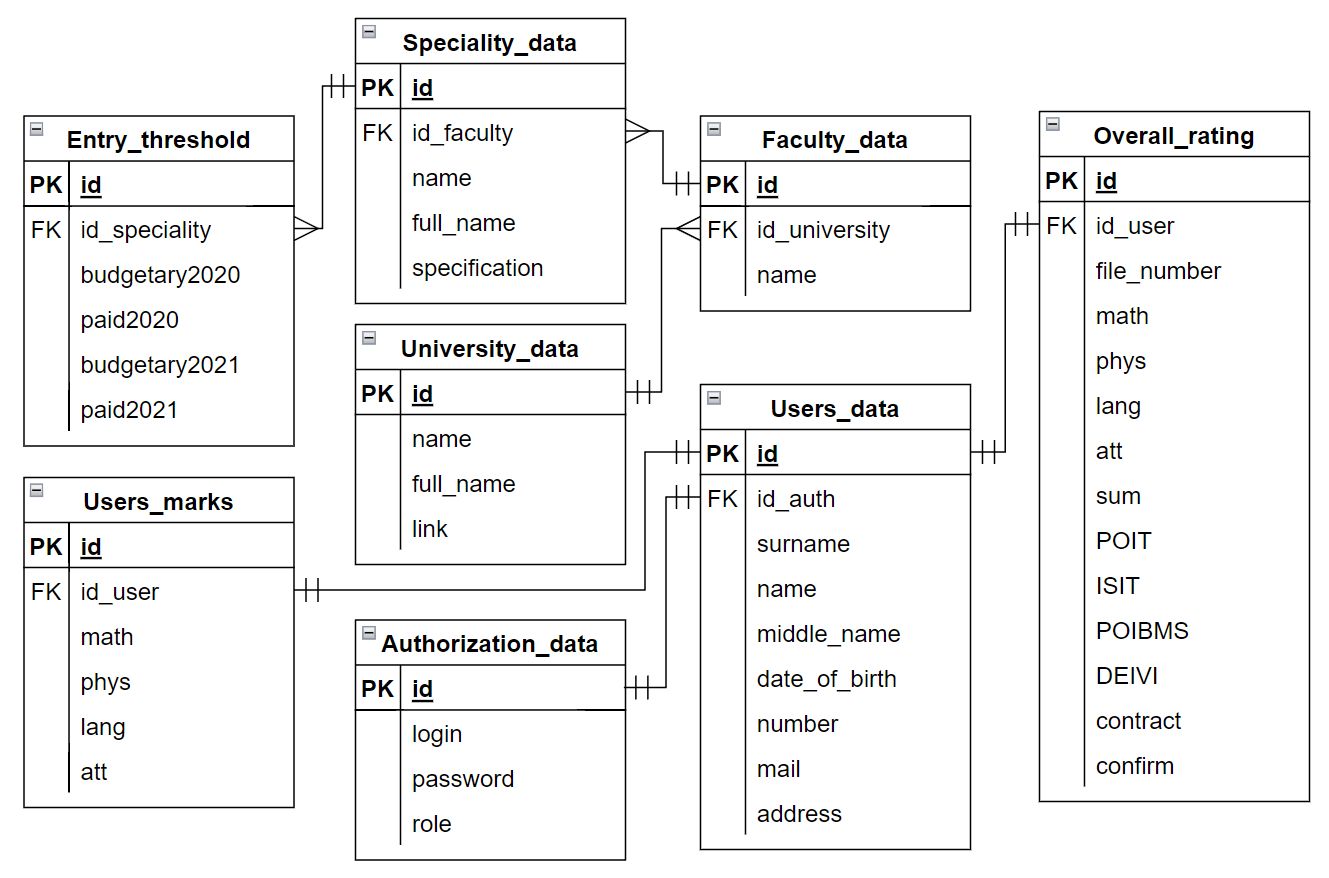


Рисунок 2.1 – Схема базы данных.

Таблица university\_data содержит данные о каждом универе. Перечень полей таблицы university\_data приведен в таблице 2.1

Таблица 2.1 – Описание полей таблицы university\_data

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | int identity primary key |
| name | varchar(255) not null |
| full\_name | varchar(255) not null |
| link | varchar(255) not null |

Охарактеризуем каждое поле таблицы university\_data:

* id – отвечает за идентификацию университета;
* name – хранит название учреждения в краткой форме;
* full\_name – хранит название учреждения в полной форме;
* link – хранит ссылку на сайт этого учреждения.

Таблица faculty\_data хранит данные о каждом факультете, каждому университету принадлежит несколько строк в этой таблице, то есть отношение такой таблицы с таблицей university\_data один ко многим.

Перечень полей таблицы faculty\_data приведен в таблице 2.2

Таблица 2.2 – Описание полей таблицы faculty\_data

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | int identity primary key |
| id\_university | int not null foreign key |
| name | varchar(255) not null |

Охарактеризуем каждое поле таблицы faculty\_data:

* id – идентификатор строки, является первичным ключом таблицы;
* id\_university – является вторичным ключом для связи с таблицей university\_data;
* name – хранит название факультета.

Таблица speciality\_data, хранит данные о каждой специальности, каждому факультету принадлежит несколько строк в этой таблице, то есть отношение такой таблицы с таблицей faculty\_data один ко многим.

Перечень полей таблицы speciality\_data приведен в таблице 2.3

Таблица 2.3 – Описание полей таблицы speciality\_data

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | int identity primary key |
| id\_faculty | int not null foreign key |
| name | varchar(255) not null |
| full\_name | varchar(255) not null |
| specification | varchar(255) not null |

Охарактеризуем каждое поле таблицы speciality\_data:

* id – идентификатор строки, является первичным ключом таблицы;
* id\_faculty – является вторичным ключом для связи с таблицей faculty\_data;
* name – хранит краткое название специальности;
* full\_name – хранит полное название специальности;
* specification – хранит информацию о спецификации специальности.

Таблица entry\_threshold хранит информацию о каждой специальности, а точнее проходные балы прошлых лет, каждой специальности принадлежит несколько строк в этой таблице, то есть отношение такой таблицы с таблицей speciality\_data один ко многим.

Перечень полей таблицы entry\_threshold приведен в таблице 2.4

Таблица 2.4 – Описание полей таблицы entry\_threshold

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | int identity primary key |
| id\_speciality | int not null foreign key |
| budgetary2020 | int not null |
| paid2020 | int not null |
| budgetary2021 | int not null |
| paid2021 | int not null |

Охарактеризуем каждое поле таблицы entry\_threshold:

* id – идентификатор строки, является первичным ключом таблицы;
* id\_speciality – является вторичным ключом для связи с таблицей speciality \_data;
* budgetary2020 – хранит проходной балл на специальность в 2020 году на бюджетной основе;
* paid2020 – хранит проходной балл на специальность в 2020 году на платной основе;
* budgetary2021 – хранит проходной балл на специальность в 2021 году на бюджетной основе;
* paid2021 – хранит проходной балл на специальность в 2021 году на платной основе.

Таблица authorization\_data хранит информацию пользователей, необходимую для аутентификации.

Перечень полей таблицы authorization\_data приведен в таблице 2.5

Таблица 2.5 – Описание полей таблицы authorization\_data

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | int identity primary key |
| login | varchar(255) not null |
| password | varchar(255) not null |
| role | varchar(255) not null |

Охарактеризуем каждое поле таблицы authorization\_data:

* id – идентификатор строки, является первичным ключом таблицы;
* login – хранит логин зарегистрированного пользователя;
* password – хранит пароли пользователей в виде строки, которая представляет собой хеш реального пароля;
* role – роль пользователя, имеет значение по умолчанию «ENROLLEE».

Таблица users\_data, хранит личные данные каждого пользователя, каждому аутентифицированному пользователю принадлежит только одна строка в этой таблице, то есть отношение такой таблицы с таблицей authorization\_data один к одному.

Перечень полей таблицы users\_data приведен в таблице 2.6

Таблица 2.6 – Описание полей таблицы users\_data

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | int identity primary key |
| id\_auth | int not null foreign key |
| surname | varchar(255) not null |
| name | varchar(255) not null |
| middle\_name | varchar(255) not null |
| date\_of\_birth | date not null |
| number | varchar(255) not null |
| mail | varchar(255) not null |
| address | varchar(255) not null |

Охарактеризуемкаждое поле таблицы userinfo:

* id – идентификатор строки, является первичным ключом таблицы;
* id\_auth – является вторичным ключом для связи с таблицей authorization\_data;
* surname, name, middle\_name – хранят в общем ФИО пользователя;
* date\_of\_birth – дату рождения пользователя;
* number – номер телефона пользователя;
* mail – адрес электронной почты пользователя;
* address – адрес пользователя.

Таблица users\_marks, хранит данные каждого пользователя о полученных им результатами ЦТ, каждому аутентифицированному пользователю принадлежит только одна строка в этой таблице, то есть отношение такой таблицы с таблицей user\_data один к одному.

Перечень полей таблицы users\_marks приведен в таблице 2.7

Таблица 2.7 – Описание полей таблицы users\_marks

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | int identity primary key |
| id\_user | int not null foreign key |
| math | int not null |
| phys | int not null |
| lang | int not null |
| att | int not null |

Охарактеризуем каждое поле таблицы users\_marks:

* id – идентификатор строки, является первичным ключом таблицы;
* id\_user – является вторичным ключом для связи с таблицей user\_data;
* math, phys, lang, att – хранит баллы пользователя по математике, физике, языку и балл аттестата.

Таблица overall\_rating, хранит данные заявок пользователей на поступление, по которым формируется мониторинг и основной конкурс поступающих, одному аутентифицированному пользователю принадлежит только одна строка в этой таблице, то есть отношение такой таблицы с таблицей user\_data один к одному.

Перечень полей таблицы overall\_rating приведен в таблице 2.8

Таблица 2.8 – Описание полей таблицы overall\_rating

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | int identity primary key |
| id\_user | int not null foreign key |
| file\_number | varchar(255) not null |
| math, phys, lang, att | int not null |
| sum | int not null |
| POIT | int not null |
| ISIT | int not null |
| POIBMS | int not null |
| DEIVI | int not null |
| contract | varchar(255) not null |
| confirm | bit not null |

Охарактеризуем каждое поле таблицы users\_marks:

* id – идентификатор строки, является первичным ключом таблицы;
* id\_user – является вторичным ключом для связи с таблицей user\_data;
* file\_number – хранит номер личного дела, который прикрепляется членом приёмной комиссии для удобства работы;
* math, phys, lang, att – хранит баллы пользователя, которые были указаны, при подаче заявки по математике, физике, языку и балл аттестата;
* sum – хранит сумму баллов при подаче заявки;
* POIT, ISIT, POIBMS, DEIVI – хранит уровень приоритета выбранной специальности, где от 1 до 4 по убыванию указывается желание, а 0 означает, что специальность не рассматривается;
* contract – хранит тип договора, принимая по стандарту одно из двух (budgetary/paid) значений;
* confirm – флаг, который устанавливается в связи с одобрением членом приёмной комиссии.

## 2.2 Идентификация и авторизация

Для выполнения задач при разработке приложения, существует необходимость регистрации, авторизации и идентификации, для этой задачи используется JWT.

JSON Web Token (JWT) — это открытый стандарт для создания токенов доступа, основанный на формате JSON. Как правило, используется для передачи данных для аутентификации в клиент-серверных приложениях. Токены создаются сервером, подписываются секретным ключом и передаются клиенту, который в дальнейшем использует данный токен для подтверждения своей личности.

Состоит из трех частей: заголовка, полезной нагрузки и зашифрованных данных, которые и являются уникальными благодаря секретным ключам.

# Разработка приложения

При проектировании программного средства в качестве программной платформы для сервера был выбран Node.js. Также на стороне клиента имеется код на JavaScript

## 3.1 Реализация базы данных

В разрабатываемом программном средстве существует возможность добавления, изменения данных в базе данных, взаимодействие происходит, как через запросы, так и через установленное соединение через веб-сокеты.

Код классов представлен на рисунке 3.1

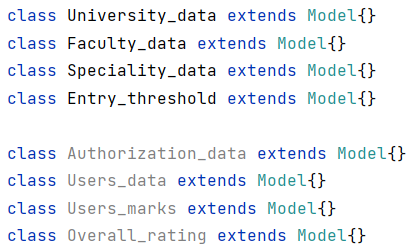


Рисунок 3.1 – Классы сущностей для ORM Sequalize.

Весь код описания сущностей базы данных представлен в приложении А.

## 3.2 Реализация серверной части.

При на писании сервера на NodeJS используется фреймворк Express.

Express — это минималистичный и гибкий веб-фреймворк для приложений Node.js, предоставляющий обширный набор функций для мобильных и веб-приложений. Он использует модуль http, а также в него включены готовые абстракции для упрощения создания сервера и серверной логики. Express не мешает общей производительности приложения, т.к. представляет собой тонкий слой основных функций веб-приложений. Также в нём удобно реализована маршрутизация.

Проект разделен на несколько основных частей, которые расположены по разным директориям. На рисунке 3.2 представлена структура проекта.

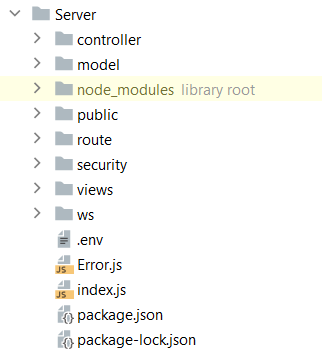


Рисунок 3.2 – Структура сервера.

В директории controller расположены контроллеры, в проекте всего пять контроллеров, так как почти вся логика реализована на веб-сокетах. Содержимое директория представлена на рисунке 3.3

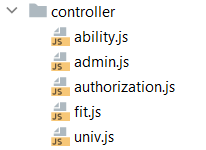


Рисунок 3.3 – Структура репозитория контроллеры.

Контроллер authorization реализует вход, выход и регистрацию в приложении, проверяют данные для регистрации при вводе в форму, проверяется логин, является ли свободным, так как наличие двух одинаковых пользователей в приложении невозможна. В файле univ, реализованы методы, для получения и обработке данных об университетах, контроллер fit описывает работу сайта университета, всё взаимодействие с личным кабинетом пользователя и мониторингом. Файл admin, описывает методы, которые позволяют администратору (члену приёмной комиссии) взаимодействовать с сервером. Контроллер ability описывает только один метод, который был необходим для тестирования, чтобы определить роль пользователя.

В папке model описан контекст базы данных, это классы для ORM и подключение к базе данных.

В папке public содержатся все статические ресурсы, необходимые для пользователя: файлы JavaScript, CSS и изображения.

В routes находятся файлы, которые сопоставляют путь, к конкретной конечной точке.

Папка security содержит файлы для https и авторизации. Для https хранится ключ и сертификат. Для JWT также два ключа для шифрования токена доступа и токена для обновления. Так же в нем описаны все привилегии и роли. Содержимое репозитория представлено на рисунке 3.4

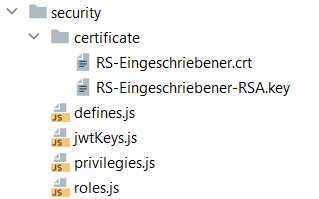


Рисунок 3.4 – Содержимое репозитория security.

Папка view содержит шаблоны и части для этих шаблонов, шаблонизатор используется handlebars. Содержимое этого репозитория представлено на рисунке 3.4

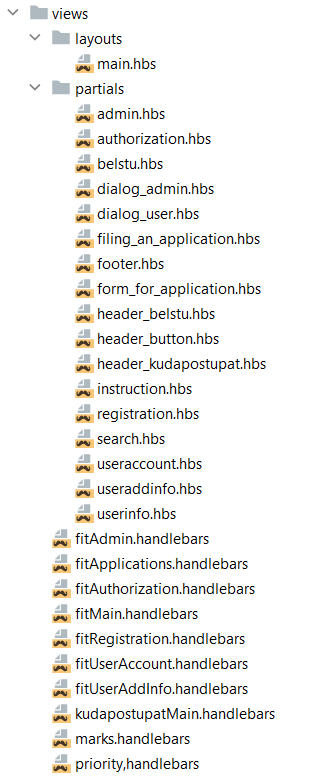


Рисунок 3.4 – Содержимое репозитория с шаблонами.

Папка ws содержит файл с веб-сокетом.

## 3.3 Веб-сокеты серверная и клиентская часть

При разработке приложения использованы веб-сокеты для поддержки диалога между клиентом и членом приёмной компании.

WebSocket — протокол связи поверх TCP-соединения, предназначенный для обмена сообщениями между браузером и веб-сервером в режиме реального времени.

Для установления соединения WebSocket клиент и сервер используют протокол, похожий на HTTP. Клиент формирует особый HTTP-запрос, на который сервер отвечает определенным образом.

Запрос от клиента на сервер представлен на рисунке 3.5

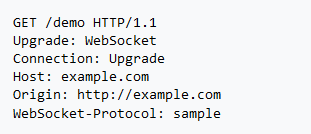


Рисунок 3.5 – Запрос клиента для установки соединения WebSocket.

Если сервер поддерживает такой протокол, то в ответ клиенту об этом, что протокол WebSocket будет использоваться для текущего соединения. Ответ можно увидеть на рисунке 3.6

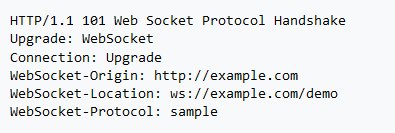


Рисунок 3.6 – Ответ от сервера, в случае успешной смены протокола.

После установки соединения по данному протоколу начинается обмен между сервером и пользователем. Сервер пересылает сообщения и другие данные для игры между игроками по этому соединению асинхронно.

Для идентификации пользователя, сервер извлекает заголовки и получает полезную нагрузку из JWT, для идентификации пользователя, для извлечения токена была реализована функция getPayLoad. Код представлен на рисунке 3.7.

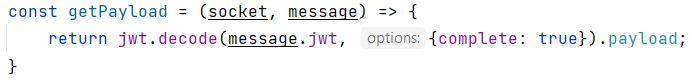


Рисунок 3.7 – Ответ от сервера, в случае успешной смены протокола.

Вначале извлекаются заголовки cookies. Код получения токена доступа из cookies представлен на рисунке 3.8



Рисунок 3.8 – Код извлечение токена доступа.

С помощью реализованного метода и осуществляется проверка, есть ли токен для доступа, если есть извлекаем полезную нагрузку из него для дальнейшей работы.

## 3.4 Проектирование клиентской части приложения

На стороне клиента используются представления, которые динамически конфигурируются с помощью движка Handlebars, который также является шаблонизатором. Он был подключён с помощью модуля handlebars-express. на рисунке 3.9

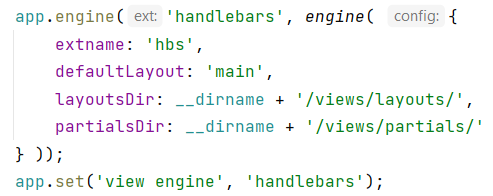


Рисунок 3.9 – Подключение движка Handlebars.

В приложении есть директория с представлениями в которой находятся все отрисованные страницы (handlebars). Также в ней есть отдельная директория, в которой представлены частичные представления, которые вставляются в другие и представление с заголовком общее для всех страниц. Также в проекте есть директорий public, который хранит в себе дополнительный функционал для клиентской части. В нём расположены стили, шрифты, файлы клиентской части кода на JavaScript, т.е. в данной папке расположены статические ресурсы моего приложения.

На рисунке 3.10 представлен пример генерирования страницы.

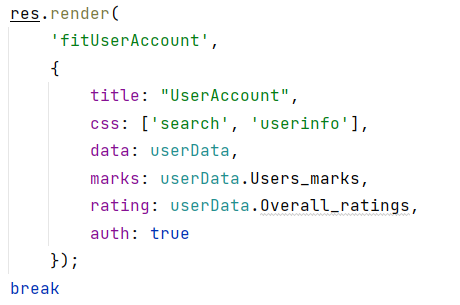


Рисунок 3.10 – Генерация страницы Handlebars.

Таким образом, генерируя страницы, можно передавать параметры, которые позволят по-разному отрисовывать страницы.

## 3.5 Конечные точки

Для клиентской части приложения была разработана серверная часть.

Описание доступных запросов взаимодействия приведено в таблице 3.1. Для взаимодействия с частью запросов необходима авторизация. Токен должен содержаться в заголовках запросов.

Запросы и описания роутеров представлены в приложении Б.

# Тестирование

В этой главе рассмотрим основные элементы интерфейса и протестируем интерфейс веб-приложения.

## 4.1 Страница поиска университета

Страница поиска состоит из четырёх полей ввода, результатов ЦТ и аттестата, кнопок для выбора типа договора и кнопки для подбора университета. В случае удачного поиска, появятся университеты и их факультеты. Интерфейс страницы можно увидеть на рисунке 4.1

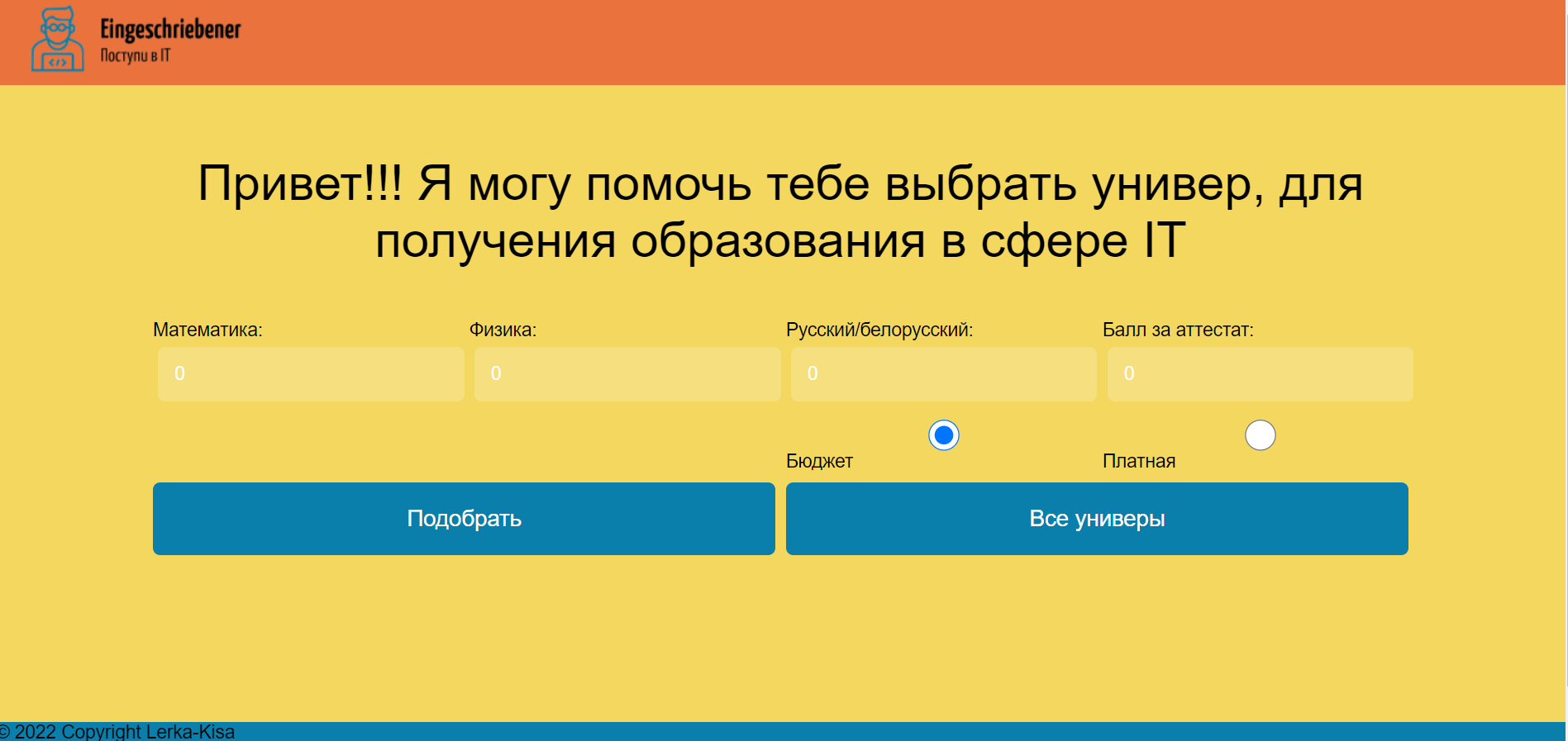


Рисунок 4.1 – Страница поиска университетов «Eingeschriebener».

При вводе результатов ЦТ по каждому предмету, проходит проверка, на то, чтоб бал не был меньше 0 или же больше 100, т.к. это невозможно, но если всё же пользователь введёт, то появиться ошибка, результат на рисунке 4.2

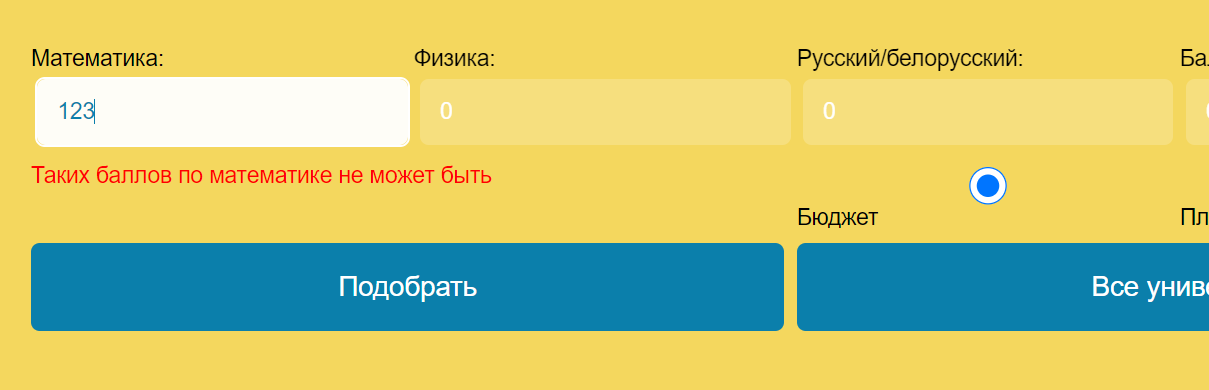


Рисунок 4.2 – Результат ввода ошибочных значений баллов.

А если он проигнорирует сообщение ошибки и попробует сделать запрос, то появится общая ошибка, результат на рисунке 4.3

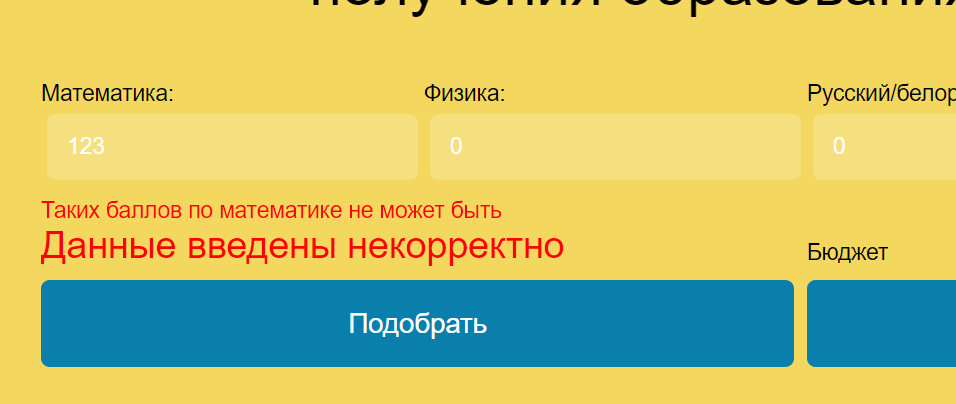


Рисунок 4.3 – Результат запроса с ошибочной информацией.

## 4.2 Форма входа сайта университета

Форма входа состоит из двух полей ввода, логина и пароля и кнопки для перехода к регистрации нового пользователя. Пользователи, которые успешно прошли поверку будут переадресованы в игру. Интерфейс формы входа можно увидеть на рисунке 4.4



Рисунок 4.4 – Форма входа в приложение «Eingeschriebener».

При вводе правильного логина и пароля пользователь перенаправляется в личный кабинет, но если там информации нет, то перенаправляется на страницу заполнения личной информации, в случае неверного входа, в поле зрения появляется текст, указывающий на ошибку, сигнализируя о неправильном вводе логина или пароля, такой пример показан на рисунке 4.5.

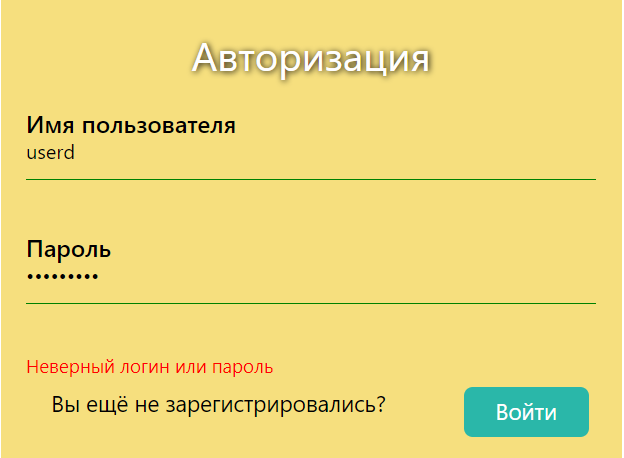


Рисунок 4.5 – Часть формы входа, при неуспешной попытке входа.

## 4.3 Форма регистрации

Форма, как можно увидеть на рисунке 4.3 состоит из пяти полей, два из которых заполняются для подтверждения ввода данных. Все поля являются обязательными для ввода. Если данные указаны неверно или пользователь уже существует, то пользователь не сможет сделать запрос для регистрации.

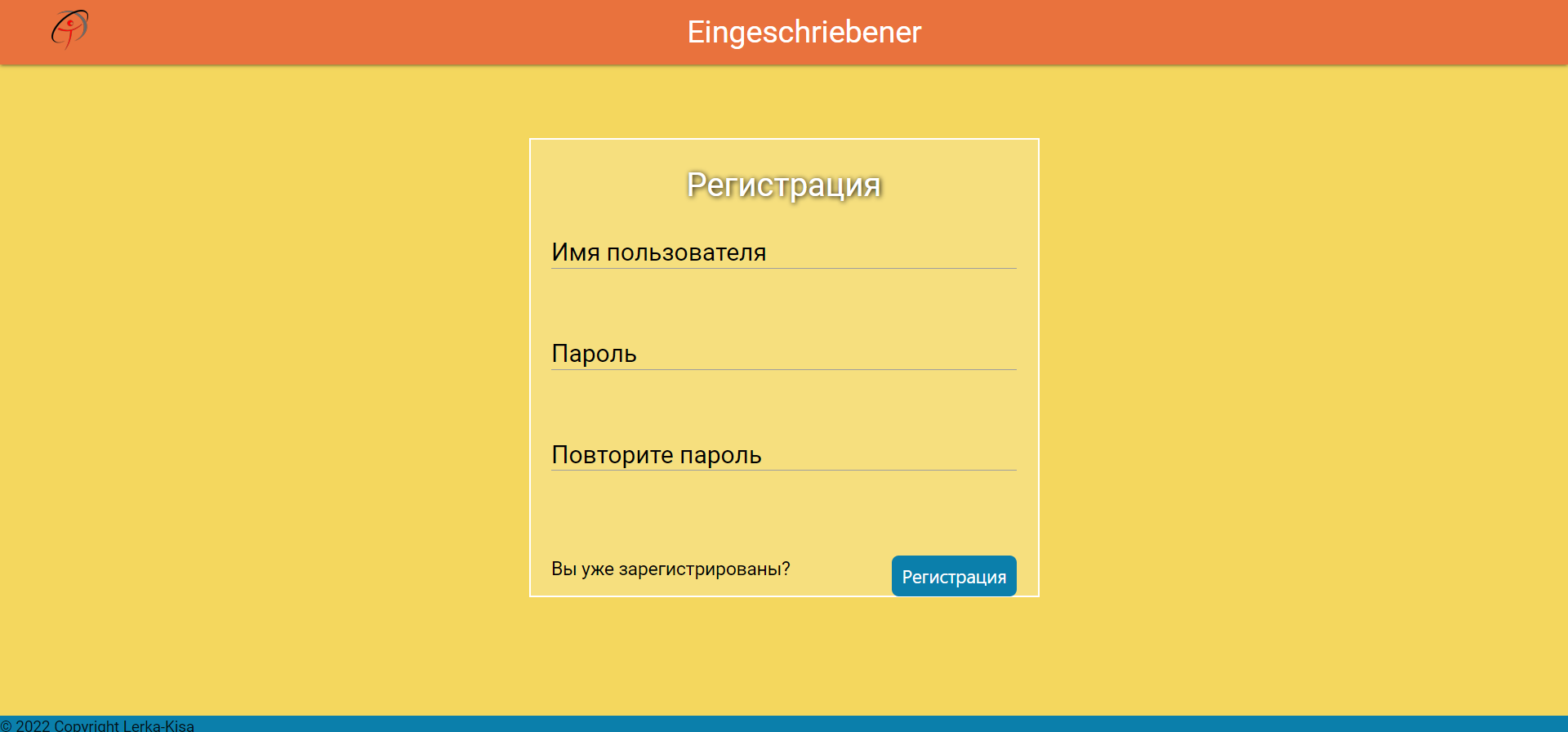


Рисунок 4.6 – Форма регистрации пользователя.

При вводе имени пользователя, в поле зрения появляется текст, указывающий на ошибку, результат такой попытке ввода, продемонстрирован на рисунке 4.7

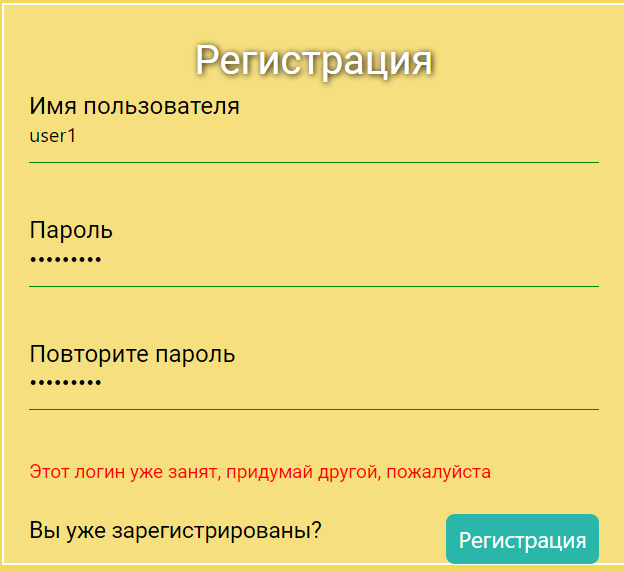


Рисунок 4.7 – Попытка ввода имени пользователя, который занят.

Так же проверяются поле пароля с полем подтверждения. Результат отображен на рисунке 4.8, при вводе разных паролей.

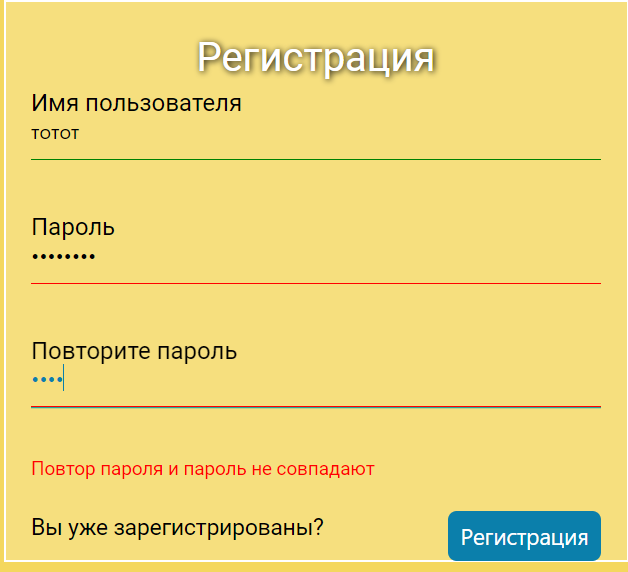


Рисунок 4.8 –Ввод в поля несовпадающих паролей.

Так же проверяется поле логина на длину введённого текста, т.к. логин не может состоять из менее чем 3 символов. Результат отображен на рисунке 4.9, при вводе разных паролей.

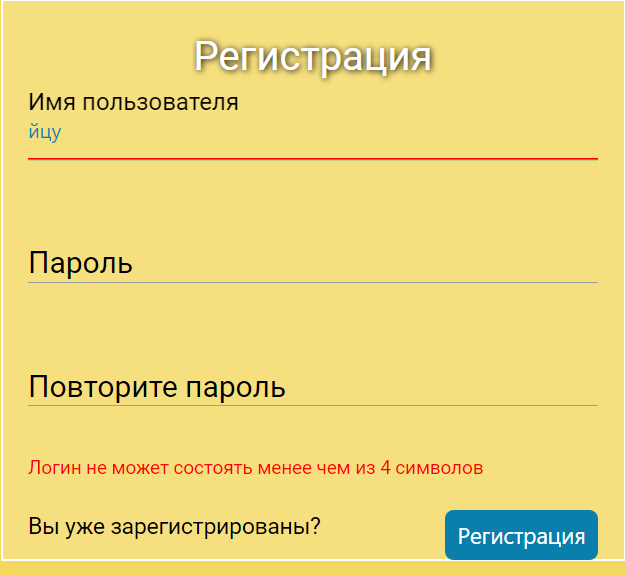


Рисунок 4.9 –Ввод в поля короткого логина.

Если пользователь исправит эти ошибки, то сообщения исчезнут.

## 4.4 Форма заполнения личной информации

Форма, как можно увидеть на рисунке 4.10 состоит из семи полей. Все поля являются обязательными для ввода. Если данные указаны неверно, то пользователь не сможет отправить.



Рисунок 4.10 – Форма заполнения личной информации.

При вводе некорректной информации, около каждого поля есть поле вывода ошибки. Пример, плохого заполнения почты и номера телефона изображены на рисунке 4.11

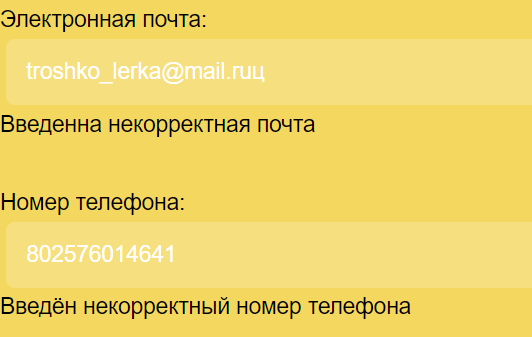


Рисунок 4.11 – Результат ошибочного заполнения почты и номера телефона.

На рисунке 4.12 продемонстрирована, работа результата запроса с ошибочными данными.

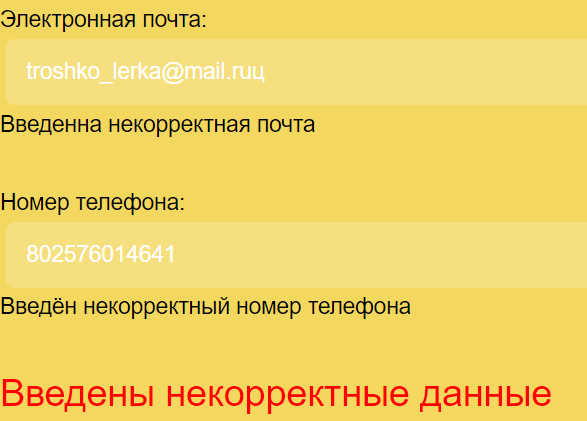


Рисунок 4.12 –Результат ошибочного запроса

В этой главе рассмотрели, основные тесты интерфейса пользователя и тестирование логики работы игры, в соответствии с техническим заданием.

# Руководство пользователя

## 5.1 Сайт для подбора университета

Первым делом пользователь попадает на главную страницу сайта, изображённую на рисунке 5.1. Как можем увидеть, в ней находятся 4 поля ввода для баллов и поле выбора контракта (бюджет/платная).

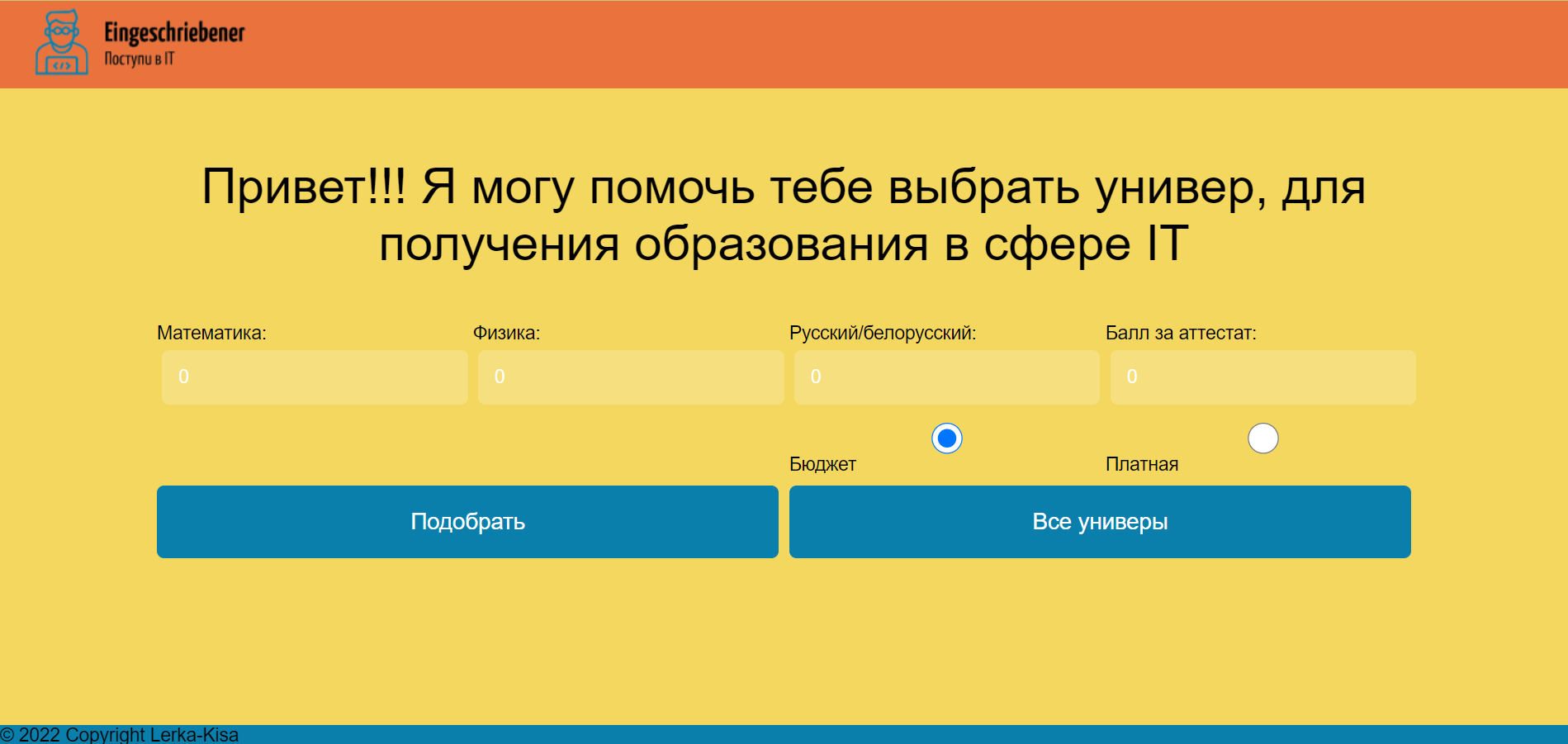


Рисунок 5.1 – Главная страница поиска университета.

При поиске университета, пользователь вводит свои баллы, они суммируются и по конкурсу прошлых лет подбираются те, в которые высокая вероятность поступить с такими баллами. При выборе «Все универы» пользователю предоставляются все университеты и их факультеты, которые позволят получить образование по IT-специальностям (рис. 5.2)

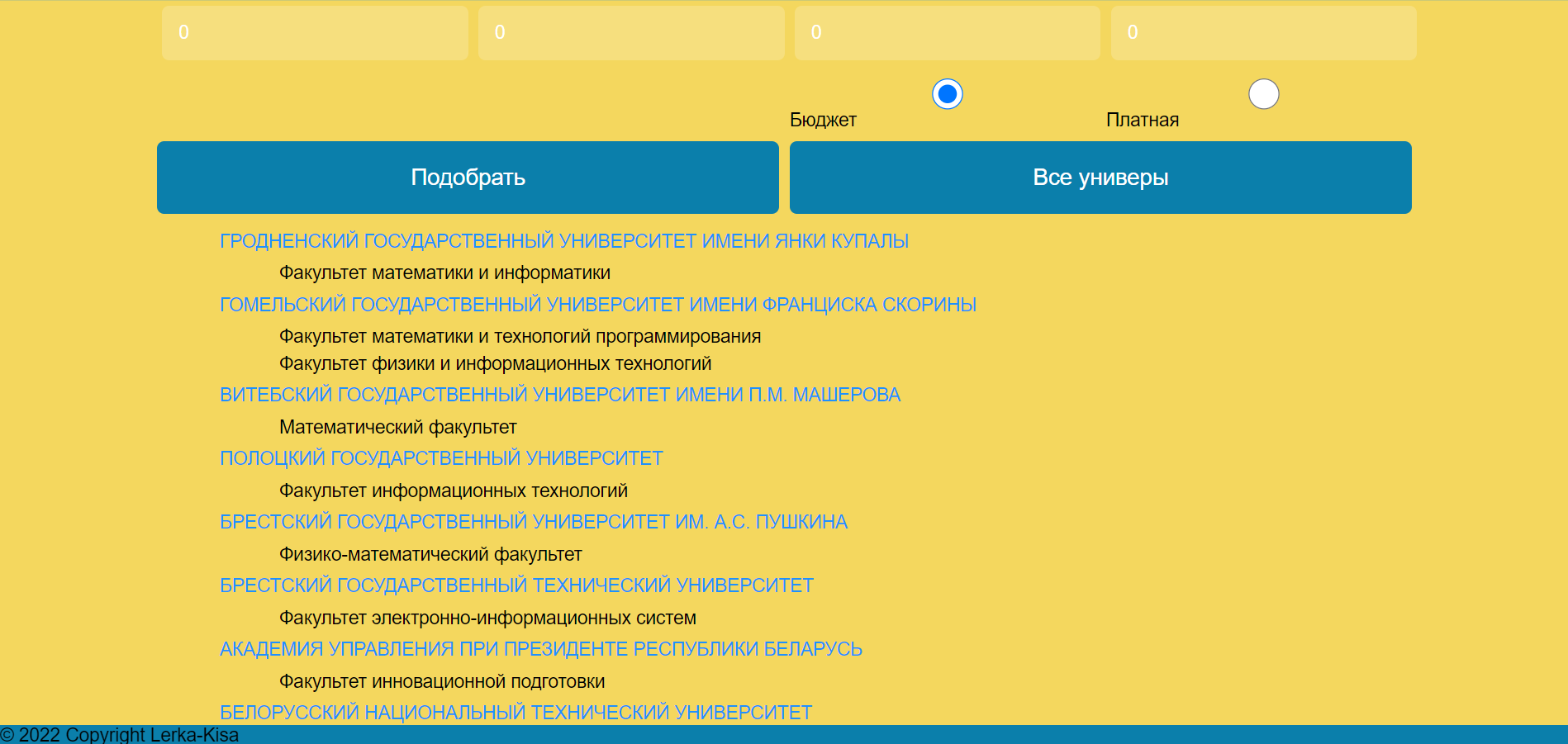


Рисунок 5.2 – Поиск университета по введённым баллам (ниже среднего).

При выборе одного из, пользователь получает страницу с более подробной информацией о факультетах, специальностях и их проходных баллах. Пример такой страницы отображён на рисунке 5.3

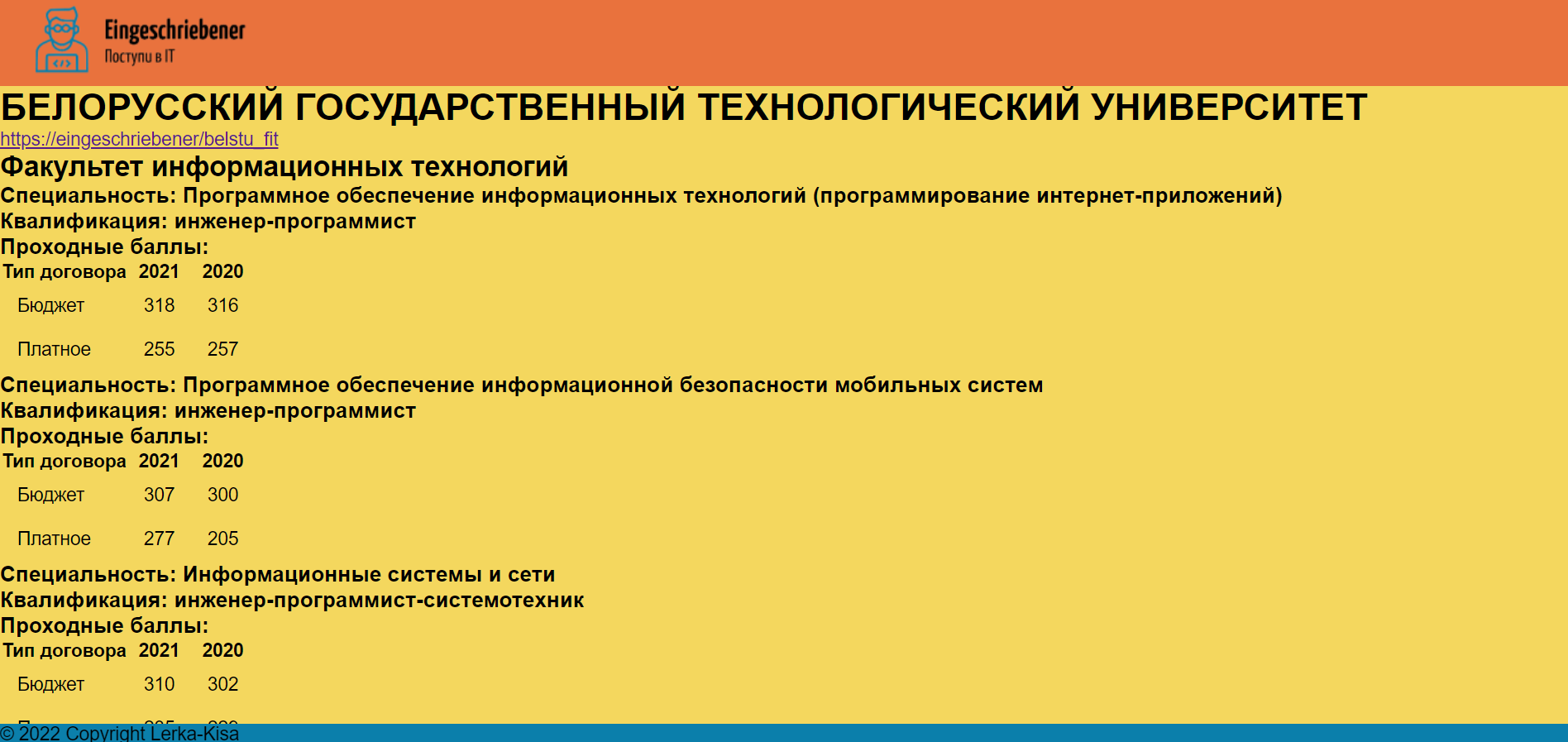


Рисунок 5.3 – Справочная информация конкретного университета

Далее, если пользователь хочет узнать более подробную информацию об университете, ему придётся перейти, по предоставленной ссылке.

## 5.2 Авторизация и регистрация

Для авторизации в приложении необходимо перейти на страницу личного кабинета или явно по адресу https://eingeschriebener/auth/login далее следует ввести логин и пароль, после чего нажать кнопку, которая изображена на рисунке 5.4, как и вся форма входа для пользователя.

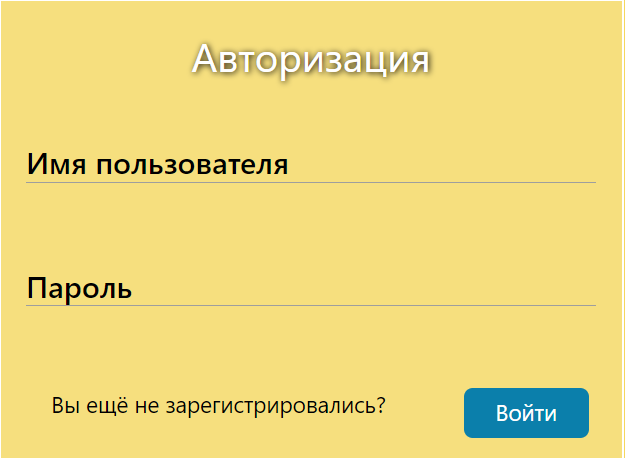


Рисунок 5.4 – Форма входа пользователя.

Для регистрации в приложении необходимо перейти на страницу входа по адресу https://eingeschriebener/auth/register*.* Форма для регистрации продемонстрирована на рисунке 5.5

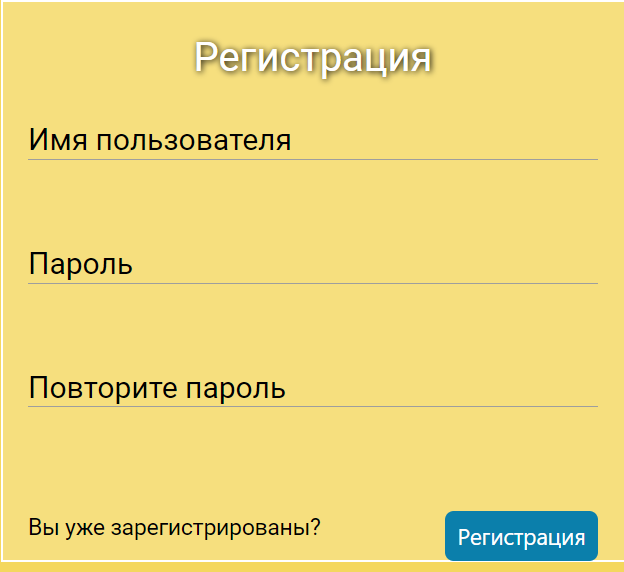


Рисунок 5.5 – Форма регистрации пользователя.

При успешной регистрации пользователь перенаправляется на форму входа.

## 5.3 Сайт университета для пользователя

Перейдя на главную страницу сайта, перед пользователем появляется краткая информация о всех специальностях, 2 общих мониторинга (платка/бюджет), для успешно авторизованного пользователя сбоку есть функция диалога с членом приёмной комиссии, в режиме реального времени, а в конце страницы есть кнопка, для подачи заявки на участие в конкурсе. Часть главной страницы сайта изображена на рисунке 5.6



Рисунок 5.6 – Главная страница сайта университета.

При успешной регистрации или идентификации, пользователь попадает в форму заполнения личной информации, изображение формы представлено на рисунке 5.7

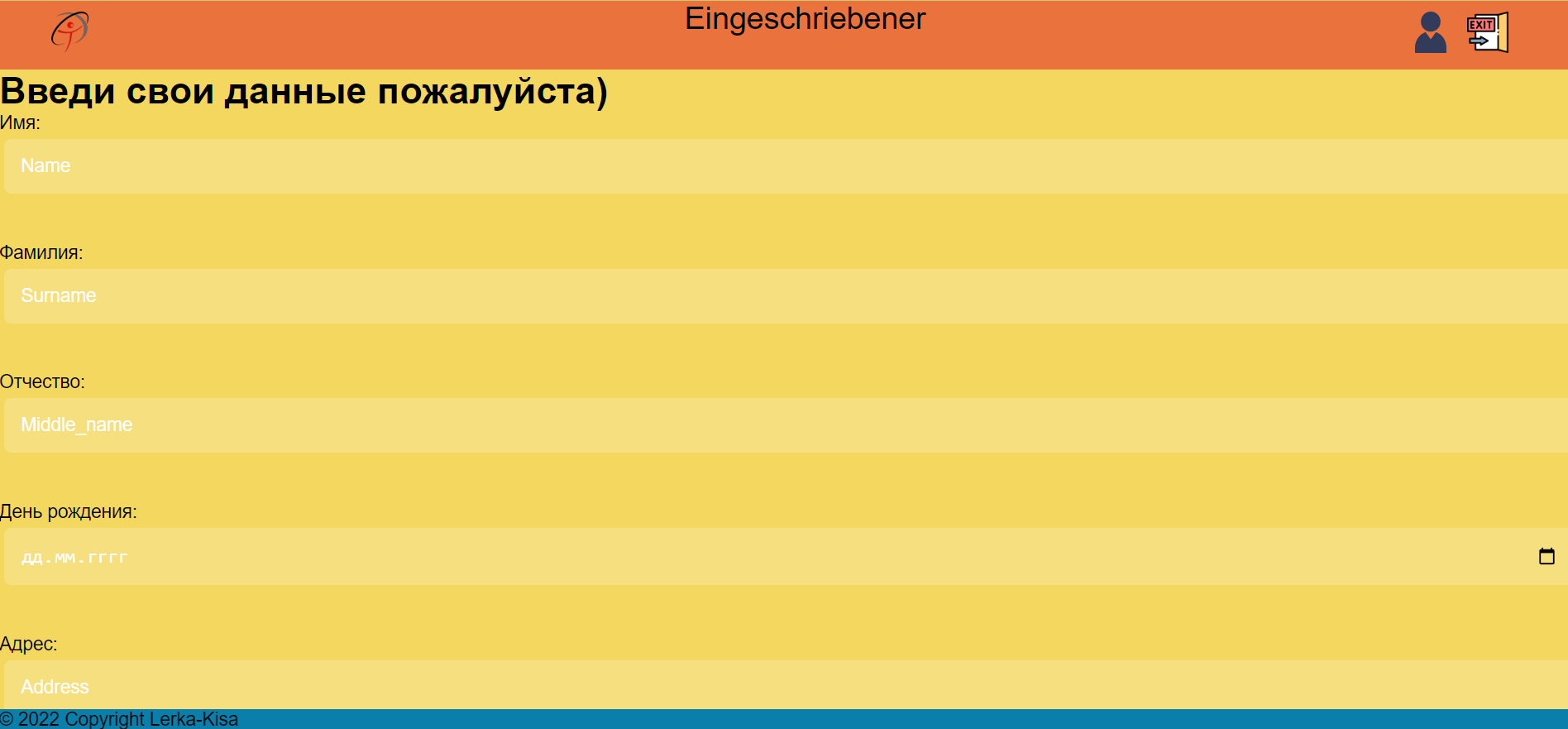


Рисунок 5.7 – Форма для добавления личной информации.

При успешном заполнении личной информации пользователь автоматически попадает в личный кабинет. Как видно на рисунке 5.8, личный кабинет состоит из 3 модулей: первый представляет личную информацию, второй – баллы пользователя и кнопку для внесения изменений в них, третий – информацию о заявки, если её нет, то пишется сообщение, если есть, то данные этой заявки и кнопка для возможности изменить её.



Рисунок 5.8 – Личный кабинет пользователя.

Для подачи заявки необходимо, чтобы у пользователя была внесена вся информация о баллах ЦТ. Для добавления этой информации необходимо пройти в личный кабинет и в модуле с баллами нажать кнопку «изменить», после чего пользователь попадает на страницу внесения изменения (рис. 5.9)

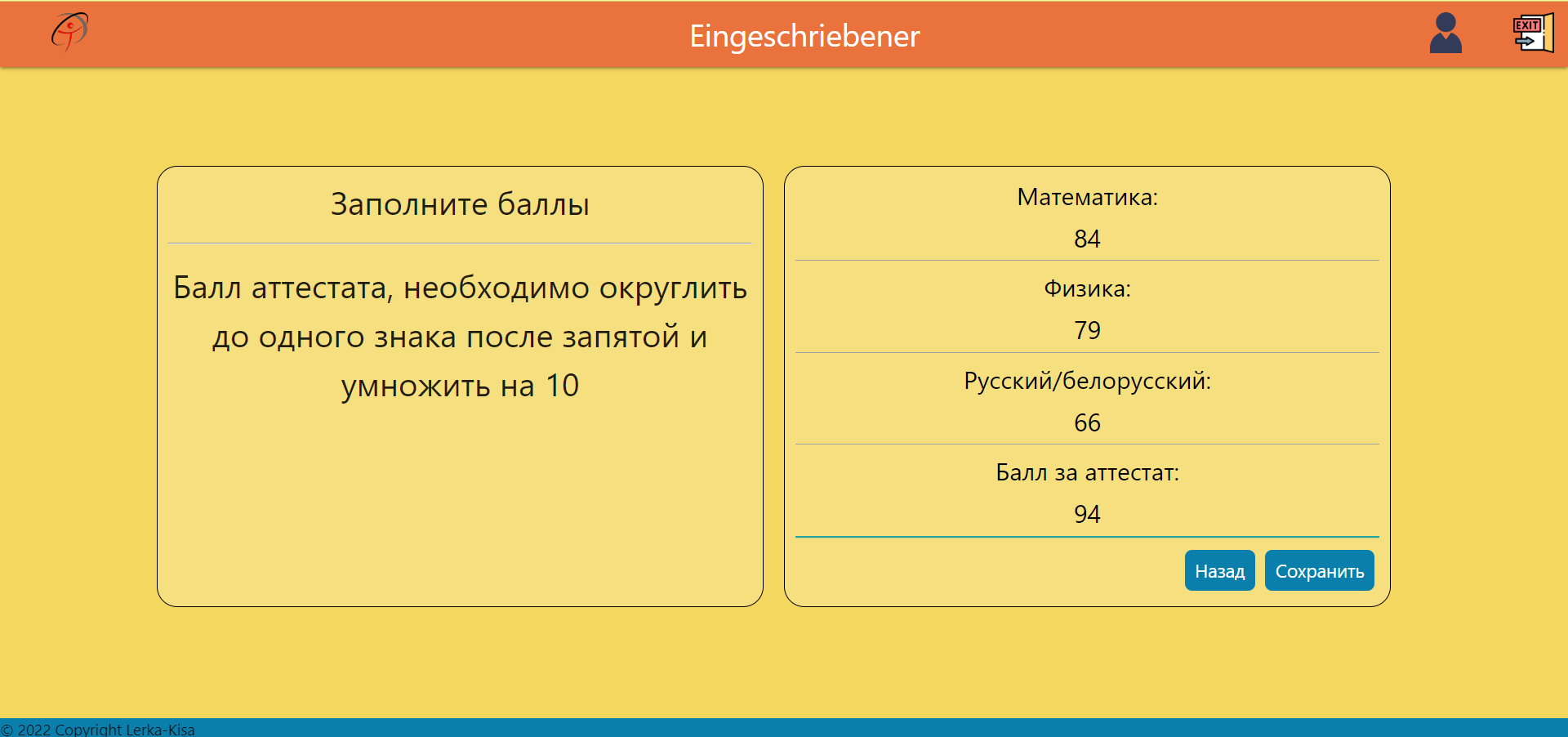


Рисунок 5.9 – Форма изменения баллов пользователя.

Если баллы по всем предметам и аттестата уже внесены, то пользователь может подать заявку на поступления, через главную страницу сайта университета. Вид заполнения заявки представлен на рисунке 5.10

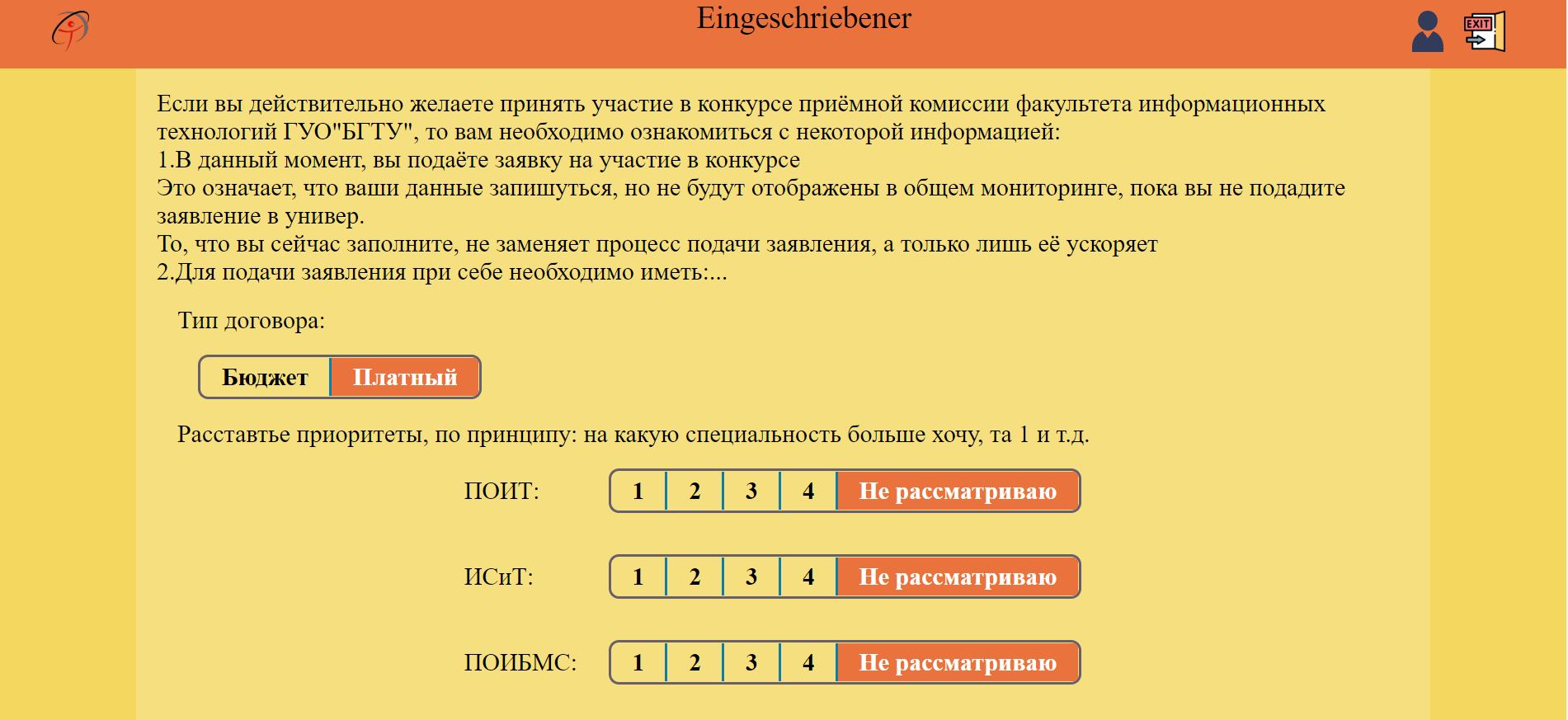


Рисунок 5.10 – Форма подачи заявки на поступление.

После отправления заявки, она отправляется на рассмотрение приёмной комиссии, т.к. эта система не заменяет процесс подачи заявления в университете, а лишь ускоряет этот процесс, то пока пользователь не принесёт необходимые документы в университет, его заявка не получит одобрения. В случае успешного прохождения подачи заявления и после одобрения заявки членом приёмной комиссии, личный пользователь может увидеть всю информацию у себя в личном кабинете (рис.5.11)



Рисунок 5.11 – Личная страница пользователя при успешно-поданной заявке.

Если пользователь решит изменить приоритеты в своей заявке, то теперь ему не надо идти в университет, а можно просто изменить её в своём личном кабинете. И попадёт на форму, изображённую на рисунке 5.12



Рисунок 5.12 –Результат ошибочного запроса

При необходимости получить консультацию, авторизованный пользователь может написать членам приёмной комиссии на главной странице сайта университета, изображение которого предоставлено на рисунке 5.13

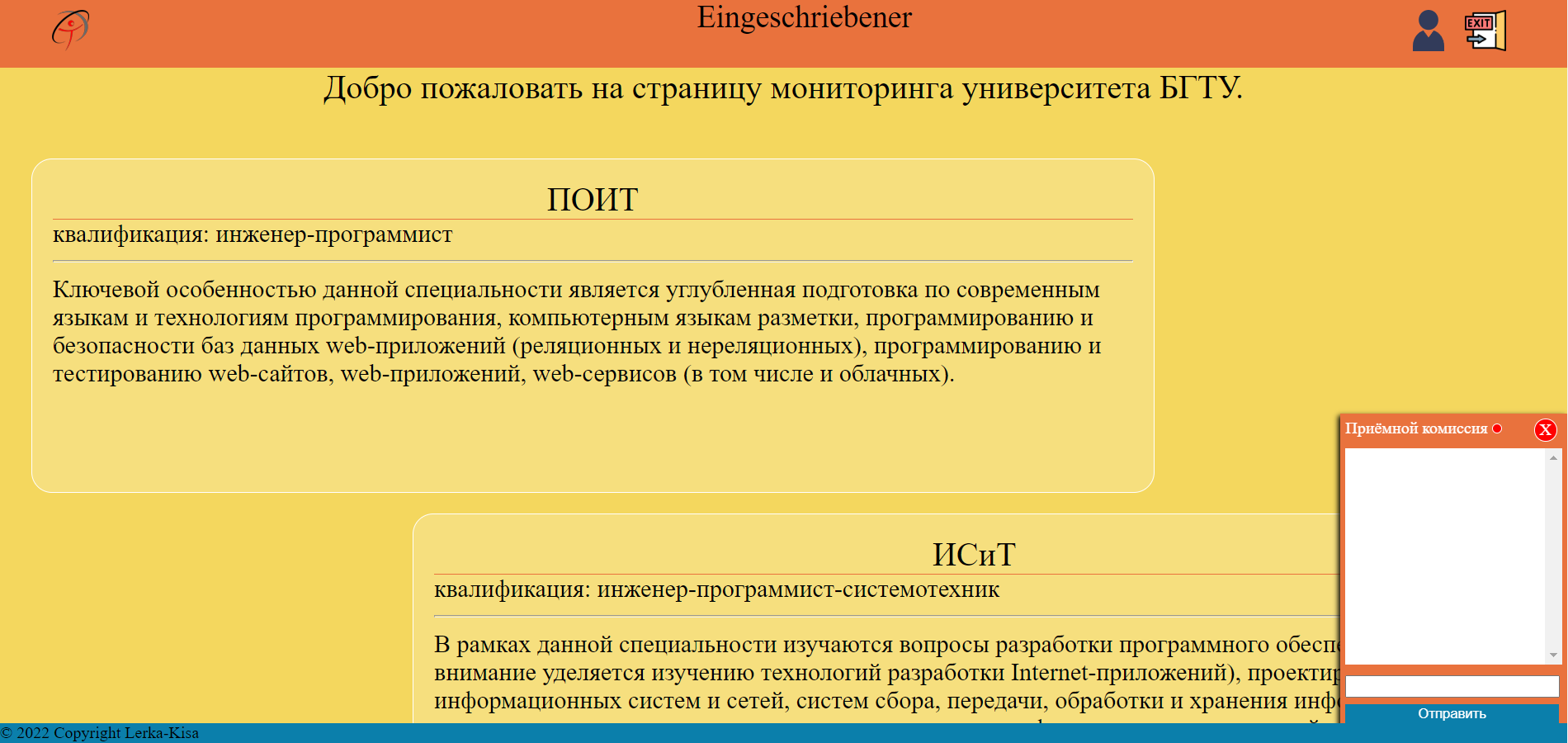


Рисунок 5.13 – Чат для общения с приёмной комиссией.

Если администратор в это время в сети, то индикатор будет гореть зелёным цветом, иначе – красным.

## 5.4 Сайт университета для члена приёмной комиссии

После авторизации администратор попадает на свою главную страницу (рис. 5.14)

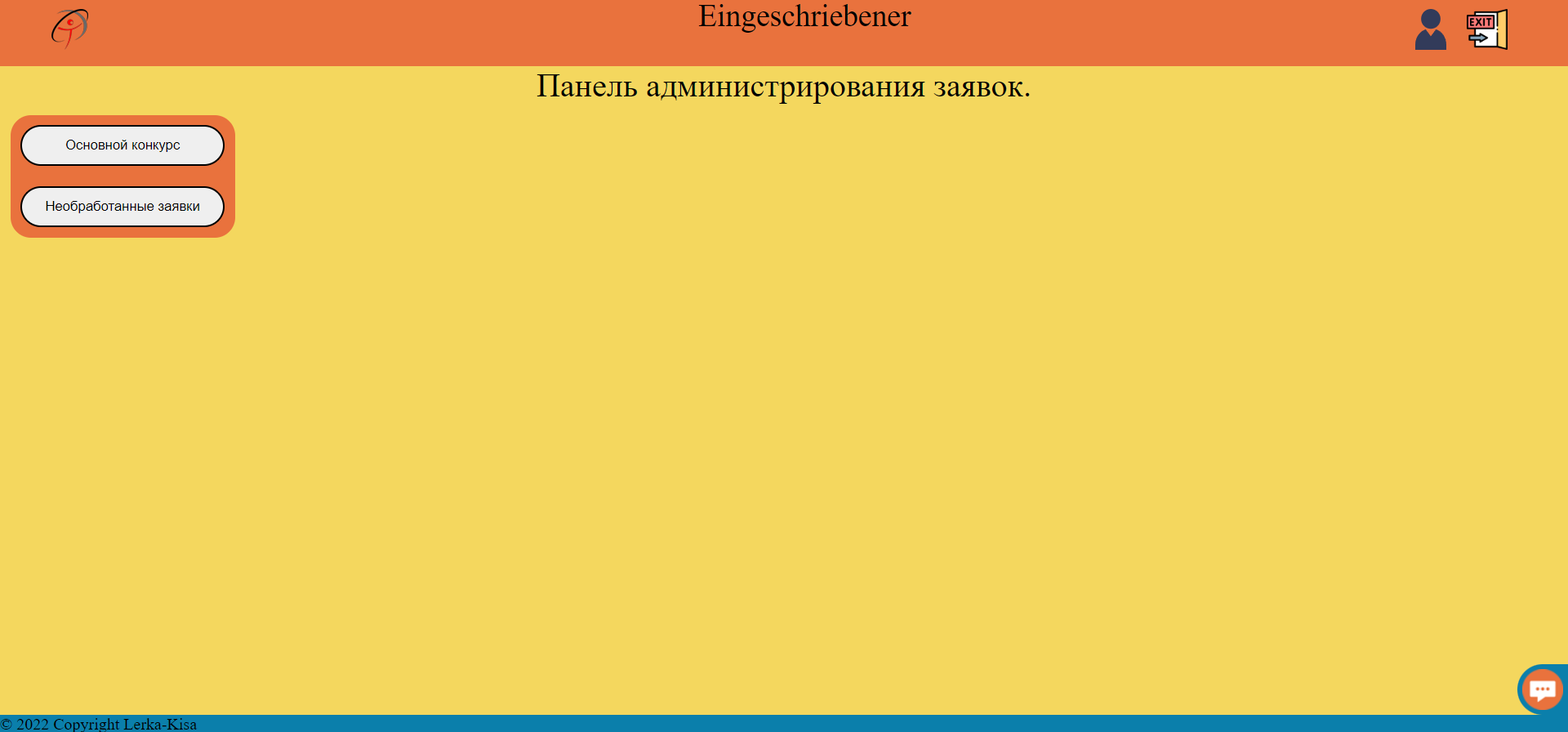


Рисунок 5.14 – Главная страница администратора

Во вкладке «Основной конкурс» отображается весь список одобренных заявок (рис. 5.15)



Рисунок 5.15 – Содержании вкладки «Основной конкурс»

Во вкладке «Необработанные заявки» (рис. 5.16), администратору предоставляется возможность получить информацию обо всех неодобренных заявках

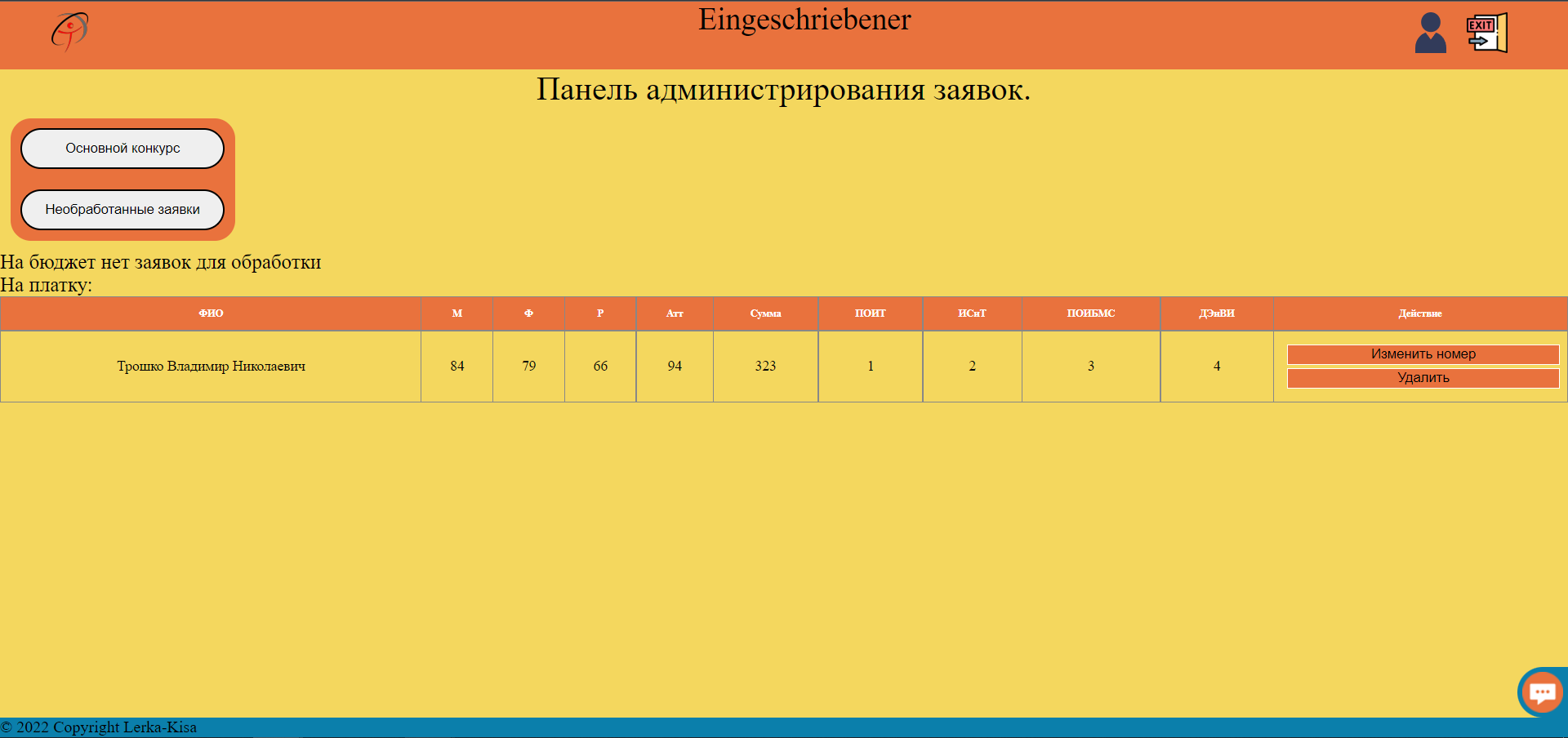


Рисунок 5.16 – Содержании вкладки «Необработанные заявки»

Кнопка для одобрения участия в общем конкурсе выводит на страницу форму для ввода номера личного дела, чтобы член приёмной комиссии мог ввести необходимые ему данные (рис. 5.17)

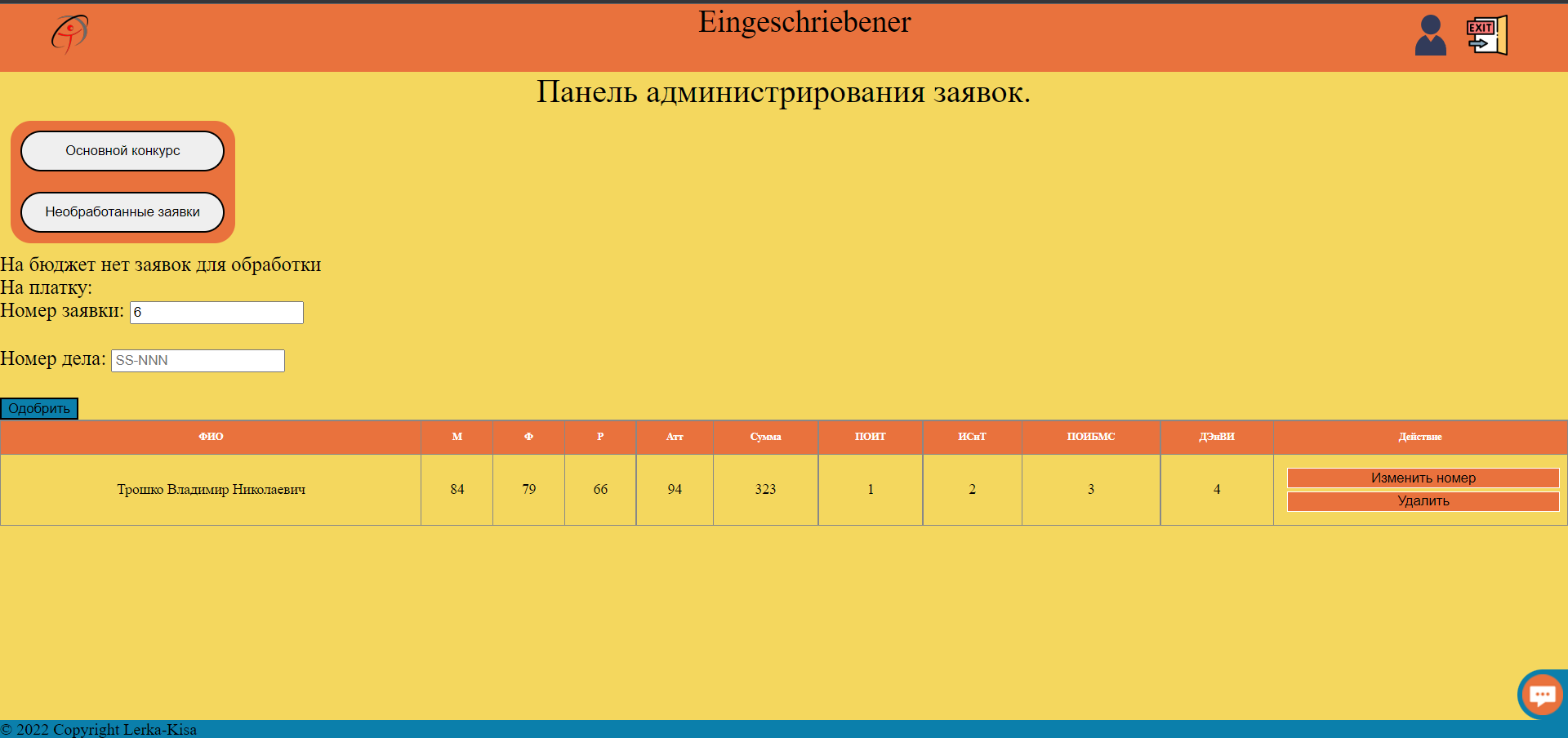


Рисунок 5.17 – Форма для внесения номера дела

При нажатии на значок сообщения справа, у пользователя появляется окно, как изображено на рисунке 5.18, с чатом, которое позволяет выбрать пользователю конкретный диалог и ответить на все существующие вопросы пользователя.

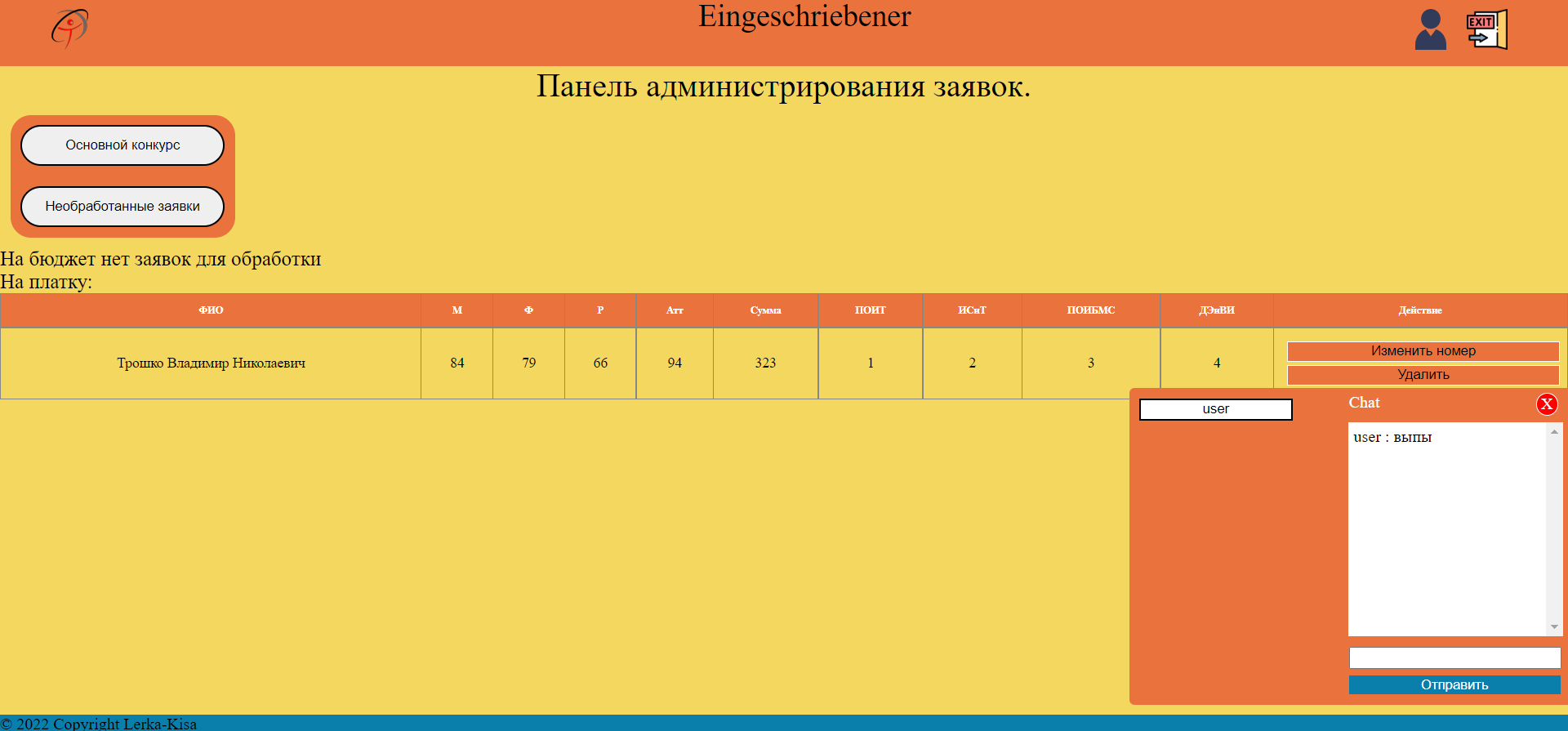


Рисунок 5.18 – Диалоги члена приёмной комиссии с абитуриентами

Как пользователь, так и член приёмной комиссии могут выйти из приложения, нажав на кнопку выхода в правом верхнем углу страницы.

# Заключение

Перед началом разработки был произведен аналитический обзор прототипов приложений подобной тематики и определение функциональных возможностей разрабатываемого приложения.

В результате написания курсового проекта, было разработано приложение, на основе архитектуры MVC. Приложение написано с помощью языка программирования Node.js. Связь между базой данных и приложением осуществляется с помощью ORM Sequelize, что упрощает работу с нашей базой данных. Также было написано руководство пользователя для веб-приложения.

Для повышения безопасности передачи данных в приложении используется HTTPS. В приложении были использованы сокеты и асинхронные запросы.

При разработке выполнены следующие задачи:

* исследованы преимущества и недостатки аналогичных приложений;
* разработана структура базы данных;
* разработана серверная часть приложения;
* разработана клиентская часть приложения;
* проведено тестирование приложения;
* написано руководство пользователя;

Представлено описание каждого этапа проектирования и реализации веб-приложения, которое позволит найти универ в сфере IT, подать заявку на поступление и отслеживание мониторинга приёмной компании онлайн.

В пояснительной записке к курсовому проекту приведено описание руководства пользователя разработанного веб-приложения.

В соответствии с полученным результатом, можно утверждать, что разработанное приложение функционирует верно, требования технического задания реализованы в полном объеме, даже больше, поэтому цель курсового проекта можно считать достигнутой.

# Список использованной литературы

1. Express [Электронный ресурс] – Режим доступа: <https://expressjs.com/ru/>– Дата доступа: 30.04.2022.
2. Express и Sequelize [Электронный ресурс] <https://metanit.com/web/nodejs/9.4.php> – Дата доступа: 02.05.2022.
3. MDN web docs Fetch API [Электронный ресурс]– Режим доступа: <https://developer.mozilla.org/ru/docs/Web/API/Fetch_API> – Дата доступа: 10.05.2022
4. Node.js [Электронный ресурс] – Режим доступа: <https://nodejs.org/en/about/> – Дата доступа: 27.04.2022.
5. Sequelize [Электронный ресурс] – Режим доступа: <https://sequelize.org/> – Дата доступа: 30.05.2022.
6. Асинхронный веб, или что такое веб-сокеты [Электронный ресурс]. — Режим доступа: <https://tproger.ru/translations/what-are-web-sockets/> Дата доступа: 17.03.2022.
7. Официальный сайт «ABITURIENT.BY» [Электронный ресурс] – Режим доступа: https://www.abiturient.by/ – Дата доступа: 23.05.2022.

Официальный сайт «I.BNTU.BY» [Электронный ресурс] – Режим доступа: https://i.bntu.by/ – Дата доступа: 23.05.2022.

# Приложение А

//Universuty.js

University\_data.init(

{

name: {type: Sequelize.STRING, allowNull:false},

full\_name: {type: Sequelize.STRING, allowNull:false},

link: {type: Sequelize.STRING, allowNull:false}

},

{sequelize, modelName:'University\_data', tableName:'University\_data', timestamps:false}

);

Faculty\_data.init(

{

id\_university: {type: Sequelize.INTEGER, allowNull:false, references: {model: University\_data, key:'id'}},

name: {type: Sequelize.STRING, allowNull:false}

},

{sequelize, modelName:'Faculty\_data', tableName:'Faculty\_data', timestamps:false}

);

Speciality\_data.init(

{

id\_faculty: {type: Sequelize.INTEGER, allowNull:false, references: {model: Faculty\_data, key:'id'}},

name: {type: Sequelize.STRING, allowNull:false},

full\_name: {type: Sequelize.STRING, allowNull:false},

specification: {type: Sequelize.STRING, allowNull:false}

},

{sequelize, modelName:'Speciality\_data', tableName:'Speciality\_data', timestamps:false}

);

Entry\_threshold.init(

{

id\_speciality: {type: Sequelize.INTEGER, allowNull:false, references: {model: Speciality\_data, key:'id'}},

budgetary2020: {type: Sequelize.INTEGER, allowNull:false},

paid2020: {type: Sequelize.INTEGER, allowNull:false},

budgetary2021: {type: Sequelize.INTEGER, allowNull:false},

paid2021: {type: Sequelize.INTEGER, allowNull:false}

},

{sequelize, modelName:'Entry\_threshold', tableName:'Entry\_threshold', timestamps:false}

);

University\_data.hasMany(Faculty\_data, {foreignKey: 'id\_university'});

Faculty\_data.belongsTo(University\_data, {foreignKey: 'id\_university'});

Faculty\_data.hasMany(Speciality\_data, {foreignKey: 'id\_faculty'});

Speciality\_data.belongsTo(Faculty\_data, {foreignKey: 'id\_faculty'});

Speciality\_data.hasMany(Entry\_threshold, {foreignKey: 'id\_speciality'});

Entry\_threshold.belongsTo(Speciality\_data, {foreignKey: 'id\_speciality'});

//Users.js

Authorization\_data.init(

{

login: {type: Sequelize.STRING, allowNull:false, unique:true},

password: {type: Sequelize.STRING, allowNull:false},

role: {type: Sequelize.STRING, allowNull:false}

},

{sequelize, modelName:'Authorization\_data', tableName:'Authorization\_data', timestamps:false}

);

Users\_data.init(

{

id\_auth: {type: Sequelize.INTEGER, allowNull:false, unique:true, references: {model: Authorization\_data, key:'id'}},

surname: {type: Sequelize.STRING, allowNull:false},

name: {type: Sequelize.STRING, allowNull:false},

middle\_name: {type: Sequelize.STRING, allowNull:false},

date\_of\_birth: {type: Sequelize.DATEONLY, allowNull:false},

number: {type: Sequelize.STRING, allowNull:false},

mail: {type: Sequelize.STRING, allowNull:false},

address: {type: Sequelize.STRING, allowNull:false}

},

{sequelize, modelName:'Users\_data', tableName:'Users\_data', timestamps:false}

);

Users\_marks.init(

{

id\_user: {type: Sequelize.INTEGER, allowNull:false, unique:true, references: {model: Users\_data, key:'id'}},

math: {type: Sequelize.INTEGER, allowNull:true},

phys: {type: Sequelize.INTEGER, allowNull:true},

lang: {type: Sequelize.INTEGER, allowNull:true},

att: {type: Sequelize.INTEGER, allowNull:true}

},

{sequelize, modelName:'Users\_marks', tableName:'Users\_marks', timestamps:false}

);

Overall\_rating.init(

{

id\_user: {type: Sequelize.INTEGER, allowNull:false, unique:true, references: {model: Users\_data, key:'id'}},

file\_number:{type: Sequelize.STRING, allowNull:false},

math: {type: Sequelize.INTEGER, allowNull:false},

phys: {type: Sequelize.INTEGER, allowNull:false},

lang: {type: Sequelize.INTEGER, allowNull:false},

att: {type: Sequelize.INTEGER, allowNull:false},

sum: {type: Sequelize.INTEGER, allowNull:false},

POIT: {type: Sequelize.INTEGER, allowNull:false},

ISIT: {type: Sequelize.INTEGER, allowNull:false},

POIBMS: {type: Sequelize.INTEGER, allowNull:false},

DEIVI: {type: Sequelize.INTEGER, allowNull:false},

contract: {type: Sequelize.STRING, allowNull:false},

confirm: {type: Sequelize.BOOLEAN, allowNull:false}

},

{sequelize, modelName:'Overall\_rating', tableName:'Overall\_rating', timestamps:false}

)

//Authorization\_data.hasOne(Users\_data, {foreignKey: 'id\_auth'});

Authorization\_data.hasMany(Users\_data, {foreignKey: 'id\_auth'});

Users\_data.belongsTo(Authorization\_data, {foreignKey: 'id\_auth'});

//Users\_data.hasOne(Users\_marks, {foreignKey: 'id\_user'});

Users\_data.hasMany(Users\_marks, {foreignKey: 'id\_user'});

Users\_marks.belongsTo(Users\_data, {foreignKey: 'id\_user'});

//Users\_data.hasOne(Overall\_rating, {foreignKey: 'id\_user'});

Users\_data.hasMany(Overall\_rating, {foreignKey: 'id\_user'});

Overall\_rating.belongsTo(Users\_data, {foreignKey: 'id\_user'});

# Приложение Б

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Адрес | Метод | | Описание | |
| Методы роутера auth | | | | |
| /login | GET | | Запрос предназначен для отображения страницы авторизации. | |
| /login | POST | | Запрос предназначен для отправки данных необходимых для авторизации. | |
| /register | GET | | Запрос предназначен для отображения страницы регистрации. | |
| / register | POST | | Запрос предназначен для отправки данных необходимых для регистрации. | |
| /logout | GET | | Запрос предназначен для выхода из аккаунта и удаления токена | |
| Методы роутера univ | | | | |
| /startpage | GET | | Запрос предназначен для отображения страницы с поиском университетов. | |
| /offer/:id | POST | | Запрос предназначен для получения информации университета по его id. | |
| /offer | GET | | Запрос предназначен для вывода всех университетов | |
| /offer | POST | | Запрос предназначен для вывода университетов с введёнными параметрами. | |
| Методы роутера fit | | | | | |
| / | | GET | | Запрос предназначен для отображения главной страницы университета. | |
| /userinfo/add | | GET | | Запрос предназначен для отображения страницы добавления личной информации пользователя. | |
| /userinfo/add | | POST | | Запрос предназначен для отправки данных необходимых для добавления личной информации пользователя. | |
| /userinfo/marks | | GET | | Запрос предназначен для получения страницы изменения баллов ЦТ абитуриента. | |
| /userinfo/marks | | POST | | Запрос предназначен для изменения баллов ЦТ абитуриента. | |
| /userinfo/change\_filing | | GET | | Запрос предназначен для получения страницы изменения приоритетов в заявке абитуриента. | |
| /userinfo/change\_filing | | POST | | Запрос предназначен для изменения приоритетов в заявке абитуриента | |
| /userinfo | | GET | | Запрос предназначен для получения страницы личного кабинета абитуриента. | |
| /getrating | | POST | | Запрос для получения информации для отображения мониторинга | |
| /check\_filing | | GET | | Запрос для разрешения на подачу заявки | |
| /filing | | GET | | Запрос предназначен для отображения страницы подачи заявки. | |
| /filing | | POST | | Запрос предназначен для регистрации заявки | |
| /admin/all\_good\_application | | POST | | Запрос предназначен для получения информации об одобренных заявках | |
| /admin/all\_bad\_application/approve | | POST | | Запрос предназначен для одобрения заявки | |
| /admin/all\_bad\_application/delete | | POST | | Запрос предназначен для отклонения заявки | |
| /admin/all\_bad\_application | | POST | | Запрос предназначен для получения информации о неодобренных заявках | |
| /admin | | GET | | Запрос предназначенный для отображения страницы администратора | |