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 асс. Кантарович В.С.

(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 асс. Кантарович В.С.

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

Норм контролёр асс. Кантарович В.С.

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

Курсовой проект защищен с оценкой

Минск 2023

Содержание

[Введение 3](#_Toc148810585)

[1 Постановка задачи 4](#_Toc148810586)

[1.1 Аналитический обзор литературы 4](#_Toc148810587)

[1.2 Анализ прототипов 6](#_Toc148810588)

[1.3 Функциональные требования 7](#_Toc148810589)

[2 Проектирование приложения 9](#_Toc148810590)

[2.1 Проектирование базы данных 9](#_Toc148810591)

[2.2 Идентификация и авторизация 15](#_Toc148810592)

[2.3 Диаграмма развёртывания 15](#_Toc148810593)

[Приложение А 16](#_Toc148810594)

[Приложение Б 20](#_Toc148810595)

[Приложение В 21](#_Toc148810596)

# 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.

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.

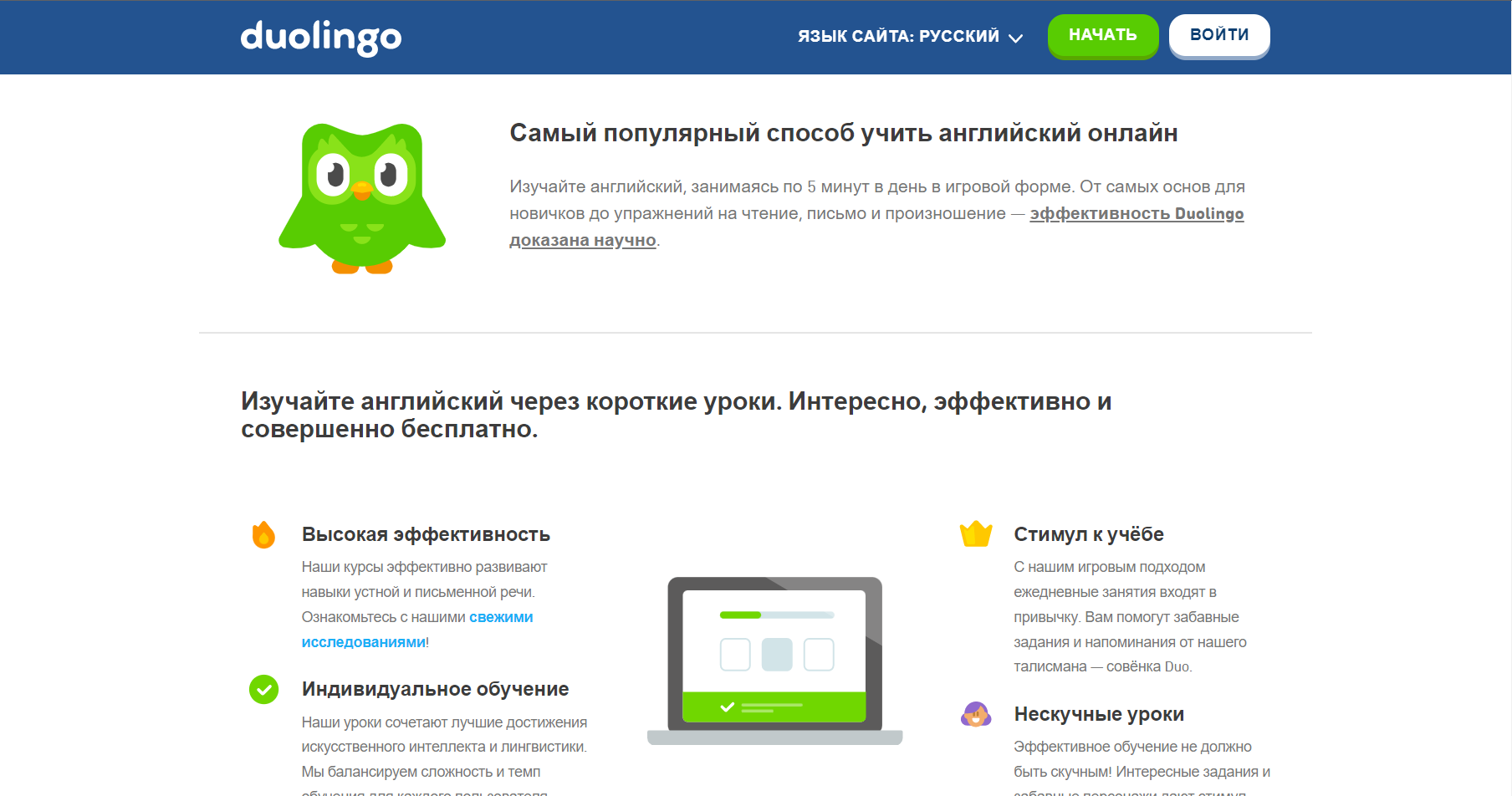


Рисунок 1.1 – Web-приложение duolingo.com

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.

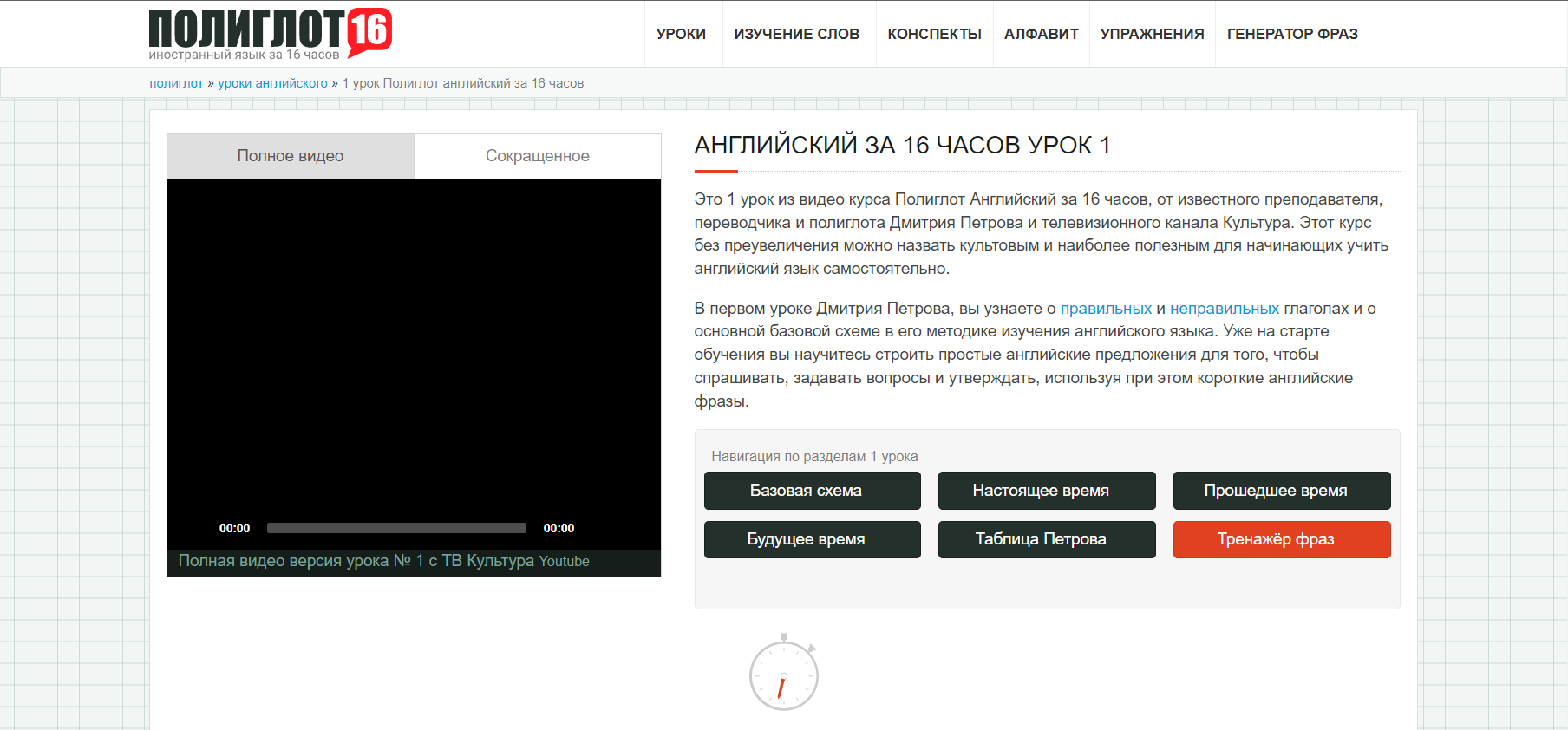


Рисунок 1.2 – Web-приложение poliglot16.ru

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.

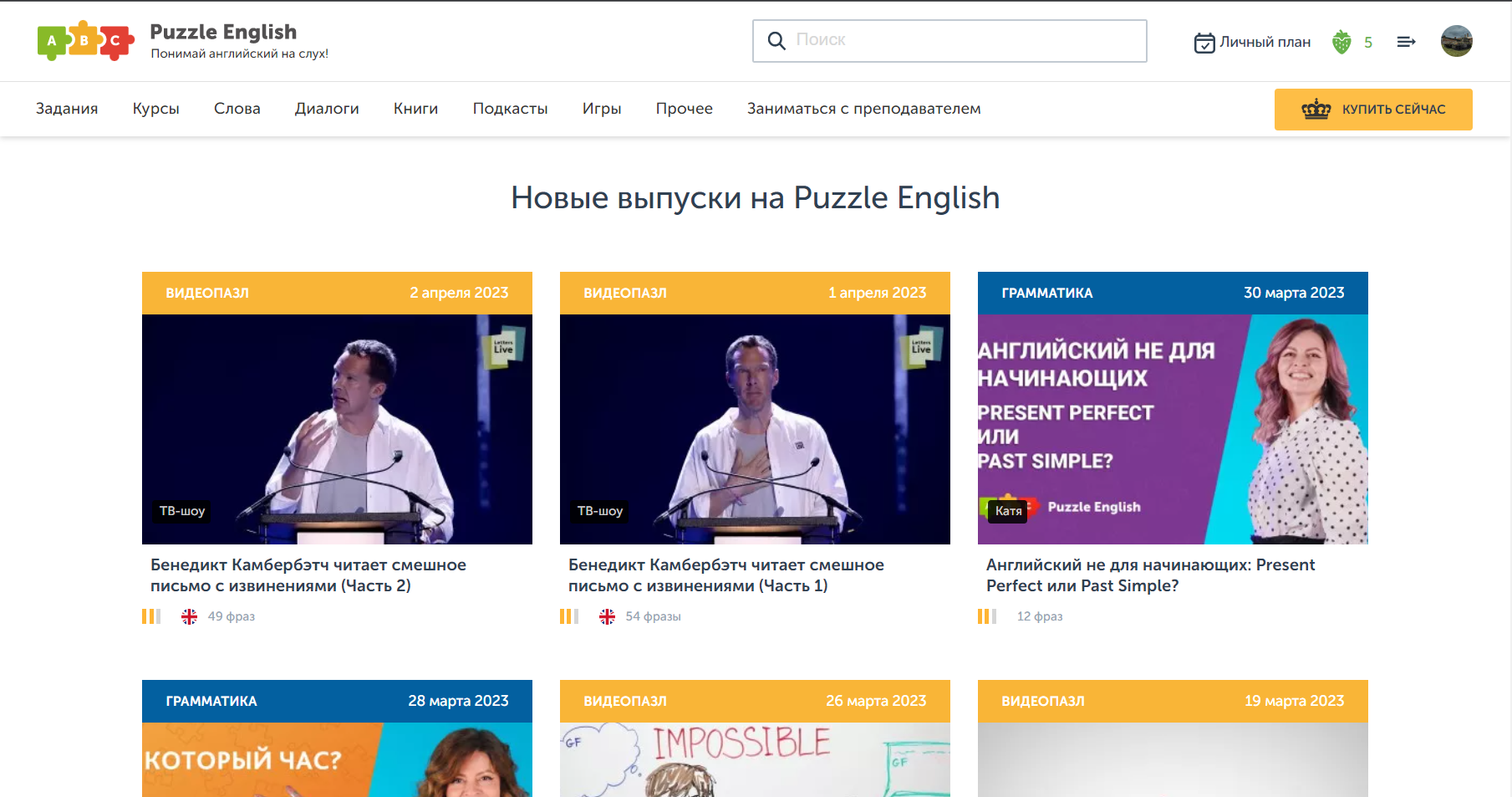


Рисунок 1.3 – Web-приложение puzzle-english.com

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.

## 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. Роль каждой учётной записи определена в базе данных. Создать администратора на сайте невозможно. Для каждой из ролей предопределены доступные страницы, поэтому при попытке получить доступ к ресурсу, которого нет или недостаточно прав, пользователь будет перенаправлен на страницу с оповещение об отсутствии искомого ресурса.

Войдя в аккаунт, пользователь получает роль авторизованного пользователя. Диаграмма вариантов использования для этой роли отображена на рисунке 1.4.

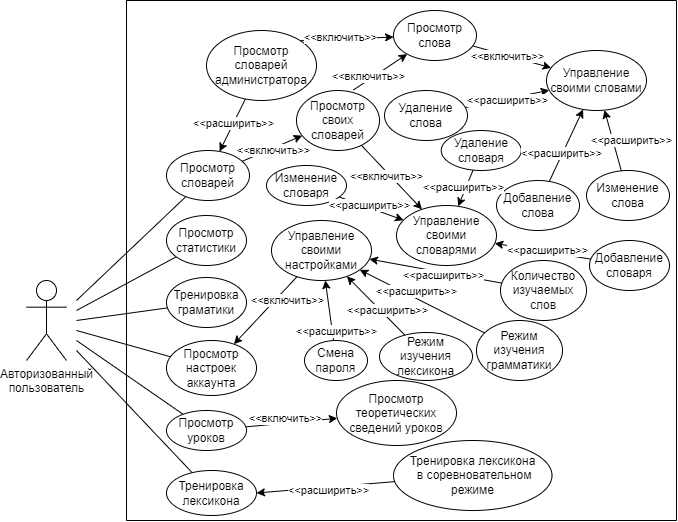


Рисунок 1.4 – Диаграмма использования авторизованного пользователя

В приложении есть роль администратора. Для этой роли диаграмма вариантов использования отображена на рисунке 1.5.

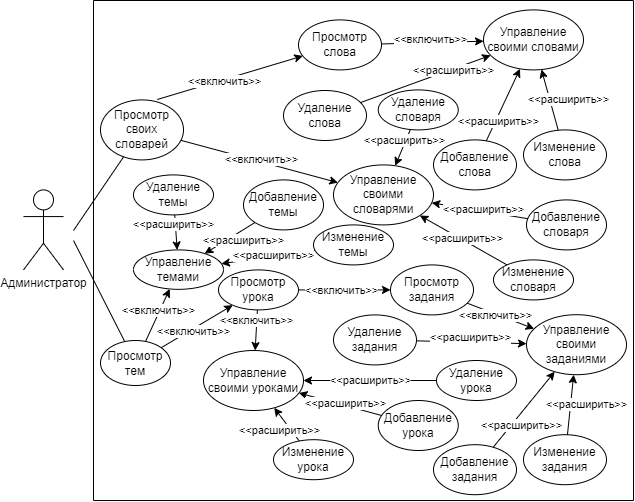


Рисунок 1.5 – Диаграмма использования администратора

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 PostgreSQL. Для взаимодействия с базой данных используется ORM Prisma.

## 2.1 Проектирование базы данных

Для реализации поставленной задачи была создана база данных посредством схемы ORM Prisma.

Прежде всего, необходимо было спроектировать корректную базу данных для работы. Схема базы данных отображена на рисунке 2.1.

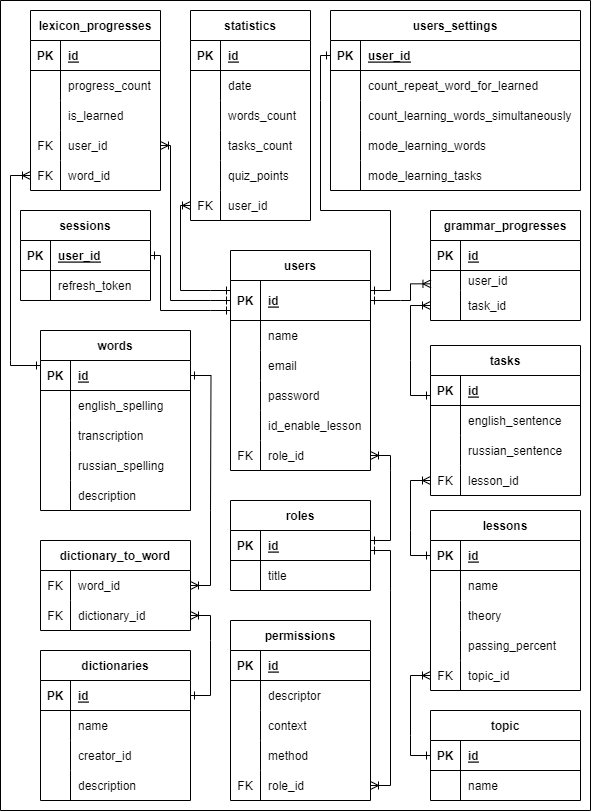


Рисунок 2.1 – Схема базы данных

Таблица dictionaries содержит словари. Перечень полей таблицы dictionaries приведен в таблице 2.1.

Таблица 2.1 – Описание полей таблицы dictionaries

|  |  |  |
| --- | --- | --- |
| Поле |  | Типы данных, ограничения целостности |
| id |  | Int, identity, primary key |
| name |  | varchar(30), not null |
| description |  | varchar(255) |
| creator\_id |  | Int, not null |

Описание каждого поля таблицы university\_data:

* id – отвечает за идентификацию словаря;
* name – название словаря;
* description – краткое описание словаря;
* creator\_id – идентификатор создателя словаря.

Таблица dictionary\_to\_word служит для связи таблиц dictionaries и words отношением многие ко многим.

Перечень полей таблицы dictionary\_to\_word приведен в таблице 2.2.

Таблица 2.2 – Описание полей таблицы dictionary\_to\_word

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| dictionary\_id | Int, not null, foreign key |
| word\_id | Int, not null, foreign key |

Описание каждого поля таблицы dictionary\_to\_word:

* dictionary\_id – внешний ключ, ссылающийся на идентификатор (id) таблицы dictionaries;
* word\_id – внешний ключ, ссылающийся на идентификатор (id) таблицы words.

Таблица grammar\_progresses служит для связи таблиц users и tasks отношением многие ко многим и каждая запись в таблице информирует о том, что определённый пользователь успешно выполнил определённое задание.

Перечень полей таблицы grammar\_progresses приведен в таблице 2.3.

Таблица 2.3 – Описание полей таблицы grammar\_progresses

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | Int, identity, primary key |
| task\_id | Int, not null, foreign key |
| user\_id | In,t not null, foreign key |

Описание каждого поля таблицы grammar\_progresses:

* id – идентификатор строки, является первичным ключом таблицы;
* task\_id – внешний ключ, ссылающийся на идентификатор (id) таблицы tasks;
* user\_id – внешний ключ, ссылающийся на идентификатор (id) таблицы users.

Таблица lessons хранит информацию о каждом уроке и имеет связь с таблицей tasks один ко многим, так как в одном уроке может быть несколько заданий.

Перечень полей таблицы lessons приведен в таблице 2.4.

Таблица 2.4 – Описание полей таблицы lessons

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | Int, identity, primary key |
| name | varchar(30), not null |
| theory | varchar(max), not null |
| passing\_percent | Int, not null |

Описание каждого поля таблицы lessons:

* id – идентификатор строки, является первичным ключом таблицы;
* name – название урока;
* theory – теоретические сведения по теме урока;
* passing\_percent – минимальный процент правильно выполненных заданий урока, для того, чтобы тема урока считалась достаточно изученной.

Таблица lexicon\_progresses служит для связи таблиц users и words отношением многие ко многим и каждая запись в таблице хранит информацию о прогрессе пользователя в изучении слов.

Перечень полей таблицы lexicon\_progresses приведен в таблице 2.5.

Таблица 2.5 – Описание полей таблицы lexicon\_progresses

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | Int, identity, primary key |
| progress\_сount | Int, not null |
| is\_learned | Boolean, not null |
| user\_id | Int, not null, foreign key |
| word\_id | Int, not null, foreign key |

Описание каждого поля таблицы lexicon\_progresses:

* id – идентификатор строки, является первичным ключом таблицы;
* progress\_сount – количество правильных переводов слова пользователем, сделанных подряд;
* is\_learned – указывает, выучено слово (true) или нет (false);
* user\_id – внешний ключ, ссылающийся на идентификатор (id) таблицы users;
* word\_id – внешний ключ, ссылающийся на идентификатор (id) таблицы words.

Таблица persmissions хранит разрешения ролей на выполнения определённых запросов.

Перечень полей таблицы persmissions приведен в таблице 2.6.

Таблица 2.6 – Описание полей таблицы persmissions

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | Int, identity, primary key |

Окончание таблицы 2.6

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| descriptor | Int, not null |
| context | varchar(255) |
| method | varchar(255), not null |
| role\_id | Int, not null, foreign key |

Описание каждого поля таблицы persmissions:

* id – идентификатор строки, является первичным ключом таблицы;
* descriptor – наименование контроллера;
* context – контекст выполняемого роута;
* method – метод http запроса;
* role\_id – внешний ключ, ссылающийся на идентификатор (id) таблицы roles.

Таблица roles хранит данные о каждой роли.

Перечень полей таблицы roles приведен в таблице 2.7.

Таблица 2.7 – Описание полей таблицы roles

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | Int, identity, primary key |
| title | varchar(255), not null |

Описание каждого поля таблицы roles:

* id – идентификатор строки, является первичным ключом таблицы;
* title – название роли.

Таблица sessions хранит данные о сессиях пользователей и связан с таблицей users связью один к одному.

Перечень полей таблицы sessions приведен в таблице 2.8.

Таблица 2.8 – Описание полей таблицы sessions

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| user\_id | Int, unique, foreign key |
| refresh\_token | varchar(max), not null, foreign key |

Описание каждого поля таблицы sessions:

* user\_id – внешний ключ, ссылающийся на идентификатор (id) таблицы users;
* refresh\_token – рефреш токен пользователя.

Таблица statistics хранит статистические данные об обучении пользователей за каждый день, а именно количество правильных переводов слов, сделанных заданий и квиз очков.

Перечень полей таблицы statistics приведен в таблице 2.9.

Таблица 2.9 – Описание полей таблицы statistics

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | Int, identity, primary key |
| date | Date, not null |
| words | int |

Окончание таблицы 2.9

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| tasks | int |
| quiz\_points | int |
| user\_id | Int, not null, foreign key |

Описание каждого поля таблицы statistics:

* id – идентификатор строки, является первичным ключом таблицы;
* date – дата создания записи;
* words – количество правильных переводов слова пользователем;
* tasks – количество правильно сделанных заданий пользователем;
* quiz\_points – количество квиз очков, полученных пользователем за правильные переводы слов в квизе;
* user\_id – внешний ключ, ссылающийся на идентификатор (id) таблицы roles.

Таблица tasks хранит задания уроков.

Перечень полей таблицы tasks приведен в таблице 2.10.

Таблица 2.10 – Описание полей таблицы tasks

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | int unique foreign key |
| english\_sentence | varchar(300) not null |
| russian\_sentence | varchar(300) not null |
| lesson\_id | int not null foreign key |

Описание каждого поля таблицы tasks:

* id – идентификатор строки, является первичным ключом таблицы;
* english\_sentence – задание, написанное на английском языке;
* russian\_sentence – перевод задания из поля english\_sentence на русский язык;
* lesson\_id – внешний ключ, ссылающийся на идентификатор (id) таблицы lesson.

Таблица users хранит данные пользователей.

Перечень полей таблицы users приведен в таблице 2.11.

Таблица 2.11 – Описание полей таблицы users

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | Int, unique, primary key |
| name | varchar(255), not null |
| email | varchar(255), not null |
| password | varchar(255), not null |
| role\_id | Int, not null, foreign key |
| id\_enable\_lesson | Int, not null |

Описание каждого поля таблицы users:

* id – идентификатор строки, является первичным ключом таблицы;
* name – имя пользователя;
* email – электронная почта пользователя;
* password – захэшированный пароль пользователя;
* role\_id – внешний ключ, ссылающийся на идентификатор (id) таблицы roles;
* id\_enable\_lesson – идентификатор (id) урока, доступного пользователю.

Таблица user\_settings хранит настройки обучения пользователей.

Перечень полей таблицы user\_settings приведен в таблице 2.12.

Таблица 2.12 – Описание полей таблицы user\_settings

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| user\_id | int unique foreign key |
| count\_repeat\_word\_for\_learned | int not null |
| count\_learning\_words\_simultaneously | int not null |
| mode\_learning\_words | varchar(255) not null |
| mode\_learning\_tasks | varchar(255) not null |

Описание каждого поля таблицы user\_settings:

* user\_id – внешний ключ, ссылающийся на идентификатор (id) таблицы users;
* count\_repeat\_word\_for\_learned – количество правильных ответов подряд, чтобы слово считалось выученным;
* count\_learning\_words\_simultaneously – количество слов изучаемы одновременно;
* mode\_learning\_words – способ изучения лексикона;
* mode\_learning\_tasks – способ изучения грамматики.

Таблица words хранит слова, добавленные администратором и пользователями.

Перечень полей таблицы words приведен в таблице 2.13

Таблица 2.13 – Описание полей таблицы words

|  |  |
| --- | --- |
| Поле | Типы данных, ограничения целостности |
| id | int unique foreign key |
| english\_spelling | int not null |
| transcription | int not null |
| russian\_spelling | varchar(255) not null |
| description | varchar(255) not null |

Описание каждого поля таблицы words:

* id – идентификатор строки, является первичным ключом таблицы;
* english\_spelling – английское написание слова;
* transcription – транскрипция слова на английском;
* russian\_spelling – русский перевод слова из поля english\_spelling;
* description – описание, необходимое для достижения однозначного понимания слова.

## 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.

## 2.3 Диаграмма развёртывания

Диаграмма развёртывания — это структурная диаграмма в языке моделирования UML, которая показывает физическое размещение компонентов программной системы на различных узлах (например, компьютеры, серверы, мобильные устройства) и связи между ними. Она используется для визуализации аппаратного и программного окружения, в котором функционирует система. На рисунке 2.2 представлена диаграмма развёртывания данного web-приложения.

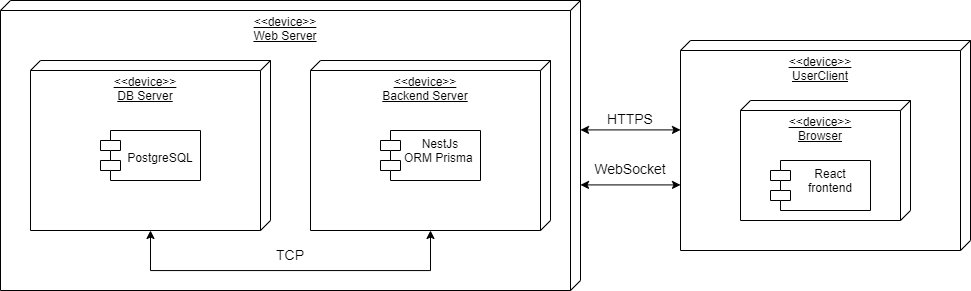


Рисунок 2.2 – Диаграмма развёртывания

На диаграмме развёртывания представлены следующие элементы:

* узлы, представляющие физические или виртуальные устройства, на которых размещаются компоненты системы (сервер, рабочая станция, мобильное устройство и т.д.);
* компоненты – это модули или части программной системы, которые могут быть развернуты на узлах (база данных, веб-сервер, клиентское приложение и т.д.);
* артефакты представляют файлы или ресурсы, необходимые для работы системы (исполняемые файлы, конфигурационные файлы, библиотеки и т.д.);
* связи показывают взаимодействия между узлами и компонентами (сетевые соединения, протоколы обмена данными и т.д.).

# Приложение А

generator client {

provider = "prisma-client-js"

}

datasource db {

provider = "postgresql"

url = env("DATABASE\_URL")

}

model User {

id Int @id @default(autoincrement())

name String

email String @unique

password String

roleId Int @default(1) @map("role\_id")

idEnableLesson Int @default(1) @map("id\_enable\_lesson")

role Role @relation(fields: [roleId], references: [id], onDelete: Cascade)

session Session?

grammarProgress GrammarProgress[]

settings UserSettings?

dictionary Dictionary[]

lexiconProgress LexiconProgress[]

statistics Statistics[]

@@map("users")

}

model UserSettings {

userId Int @unique @map("user\_id")

user User @relation(fields: [userId], references: [id], onDelete: Cascade)

countRepeatWordForLearned Int @map("count\_repeat\_word\_for\_learned")

countRepeatWordsSimultaneously Int @map("count\_learning\_words\_simultaneously")

learningModeWords LearningMode

learningModeTasks LearningMode

@@map("users\_settings")

}

model Topic {

id Int @id @default(autoincrement())

name String

lessons Lesson[]

@@map("topic")

}

model Lesson {

id Int @id @default(autoincrement())

name String

theory String

passingPercent Int @map("passing\_percent")

tasks Task[]

@@map("lessons")

}

model Task {

id Int @id @default(autoincrement())

englishSentence String @map("english\_sentence")

russianSentence String @map("russian\_sentence")

lessonId Int @map("lesson\_id")

lesson Lesson @relation(fields: [lessonId], references: [id], onDelete: Cascade)

grammarProgress GrammarProgress[]

@@unique([englishSentence, russianSentence, lessonId])

@@map("tasks")

}

model GrammarProgress {

id Int @id @default(autoincrement())

userId Int @map("user\_id")

user User @relation(fields: [userId], references: [id], onDelete: Cascade)

taskId Int @map("task\_id")

task Task @relation(fields: [taskId], references: [id], onDelete: Cascade)

@@unique([userId, taskId])

@@map("grammar\_progresses")

}

model Dictionary {

id Int @id @default(autoincrement())

name String

description String?

creatorId Int @map("creator\_id")

user User @relation(fields: [creatorId], references: [id], onDelete: Cascade)

dictionaryToWord DictionaryToWord[]

words Word[]

@@map("dictionaries")

}

model Word {

id Int @id @default(autoincrement())

englishSpelling String @map("english\_spelling")

transcription String

russianSpelling String @map("russian\_spelling")

description String?

lexiconProgress LexiconProgress[]

dictionaryToWord DictionaryToWord[]

dictionaries Dictionary[]

@@unique([englishSpelling, russianSpelling])

@@map("words")

}

model DictionaryToWord {

id Int @id @default(autoincrement())

dictionaryId Int

wordId Int

dictionary Dictionary @relation(fields: [dictionaryId], references: [id], onDelete: Cascade)

word Word @relation(fields: [wordId], references: [id], onDelete: Cascade)

@@unique([dictionaryId, wordId])

@@map("dictionary\_to\_word")

}

model LexiconProgress {

id Int @id @default(autoincrement())

progressCount Int @map("progress\_count")

isLearned Boolean @map("is\_learned")

userId Int @map("user\_id")

user User @relation(fields: [userId], references: [id], onDelete: Cascade)

wordId Int @map("word\_id")

word Word @relation(fields: [wordId], references: [id], onDelete: Cascade)

@@unique([userId, wordId])

@@map("lexicon\_progresses")

}

model Statistics {

id Int @id @default(autoincrement())

date DateTime

words Int?

tasks Int?

quizPoints Float? @map("quiz\_points")

userId Int @map("user\_id")

user User @relation(fields: [userId], references: [id], onDelete: Cascade)

@@unique([date, userId])

@@map("statistics")

}

model Session {

userId Int @unique @map("user\_id")

user User @relation(fields: [userId], references: [id], onDelete: Cascade)

refreshToken String @map("refresh\_token")

@@map("sessions")

}

model Role {

id Int @id @default(autoincrement())

title String

permissions Permission[]

users User[]

@@map("roles")

}

model Permission {

id Int @id @default(autoincrement())

descriptor String

context String?

method Method

roleId Int @map("role\_id")

role Role @relation(fields: [roleId], references: [id], onDelete: Cascade)

@@map("permissions")

}

enum Method {

ALL

GET

POST

DELETE

PUT

PATCH

}

enum LearningMode {

TRANSLATE\_FROM\_ENGLISH

TRANSLATE\_FROM\_RUSSIAN

COMBINED

}

# Приложение Б



# Приложение В

