

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Новосибирский национальный исследовательский государственный
университет»
(Новосибирский государственный университет, НГУ)
Структурное подразделение Новосибирского государственного университета –
Высший колледж информатики Университета (ВКИ НГУ)
КАФЕДРА ИНФОРМАТИКИ

Разработка

Квалификация программист

Руководитель

Фамилия Ф.Ф.

«___» 2024 г.

Студент 4 курса
гр. 007В

Фамилия Ф.Ф.
«___» Игнатов Максим
Евгеньевич 2024 г.

Новосибирск

2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	4
1 ПОСТАНОВКА ЗАДАЧИ ВКР.....	6
1.1 Бизнес-требования.....	6
1.2 Пользовательские требования.....	7
1.3 Системные требования.....	8
1.4 Требования к графическому пользовательскому интерфейсу.....	9
1.5 План-график выполнения ВКР.....	10
2 АНАЛИЗ ТРЕБОВАНИЙ И ОПРЕДЕЛЕНИЕ СПЕЦИФИКАЦИЙ....	12
2.1 Описание предметной области задачи ВКР.....	12
2.1.1 Информационные объекты предметной области и взаимосвязи между ними.....	12
2.1.2 Информационные и функциональные потребности пользователей разрабатываемой ПС.....	13
2.1.3 Методы работы с информационными объектами предметной области.....	14
2.1.4 Обзор существующих программных реализаций решения задачи	14
2.1.5 Концептуальное обоснование разработки.....	18
2.2 Классы и характеристики пользователей.....	18
2.3 Функциональные требования.....	20
1 2.3.1 Определение функциональных возможностей ПС.....	20
2.3.2 Описание прецедентов.....	23
2.4 Нефункциональные требования.....	24
3 ВЫБОР ПРОГРАММНЫХ СРЕД И СРЕДСТВ РАЗРАБОТКИ.....	27
3.1 Сравнительный анализ имеющихся возможностей по выбору средств разработки.....	27

3.2 Характеристика выбранных программных сред и средств.....	28
4 АЛГОРИТМ РЕШЕНИЯ ПОСТАВЛЕННОЙ ЗАДАЧИ.....	30
4.1 Этапы реализации ПС.....	30
4.2 Пользовательский интерфейс ПС.....	31
4.2.1 Взаимодействие пользователей с ПС.....	31
4.2.2 Проектирование пользовательских сценариев.....	32
4.2.3 Определение операций пользователей.....	34
4.2.4 Составление функциональных блоков.....	34
4.2.5 Проектирование структуры экранов ПС и схемы навигации.....	35
4.2.6 Низкоуровневое проектирование.....	36
4.3 Входные, выходные и промежуточные данные.....	39
4.4 Разработка базы данных, реализуемой в рамках ПС.....	40
4.6 Архитектура и схема функционирования ПС.....	42
5 ТЕСТИРОВАНИЕ И ОПТИМИЗАЦИЯ.....	45
5.1 План тестирования.....	45
5.2 Результаты тестирования.....	46
6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ.....	48
ЗАКЛЮЧЕНИЕ.....	51
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ.....	52
ПРИЛОЖЕНИЯ.....	53
Приложение А.....	53
Приложение Б.....	56

ВВЕДЕНИЕ

В современном мире знание английского языка является необходимым и крайне актуальным. Это обусловлено глобализацией, расширением международных связей в различных областях, таких как бизнес, образование, туризм и другие. Владение английским языком открывает перед человеком множество возможностей, включая карьерный рост, обучение за рубежом и путешествия.

Особенно важно владение английским языком для специалистов в сфере информационных технологий, поскольку значительная часть технической документации и профессиональной литературы доступна исключительно на английском языке. Таким образом, знание английского языка становится ключевым навыком для ИТ-специалистов.

Однако процесс изучения иностранного языка нередко сопровождается определёнными трудностями, особенно на начальном этапе. Необходимо запоминать большое количество новой лексики, грамматических правил, а также овладевать правильным произношением. Дополнительные сложности возникают из-за недостатка времени и возможности заниматься с преподавателем, что особенно актуально для взрослых людей, совмещающих учёбу с работой и другими обязанностями. [1]

Существующие компьютерные программы и приложения для изучения английского языка зачастую не учитывают потребности и возможности начинающих. Многие из них ориентированы на пользователей, уже обладающих определённой базой знаний.

В Высшем колледже информатики студенты начинают изучать английский язык с первого курса. Обучение включает курс базового английского языка, переходящий в курс профессионального уровня, который завершается экзаменом. Это подчеркивает необходимость создания специализированного учебного ресурса, ориентированного на студентов, осваивающих основы английского языка.

Разработка веб-приложения «IT-English» для студентов Высшего колледжа информатики, изучающих базовый уровень английского языка, является актуальной задачей. Целью данной выпускной квалификационной работы является

создание веб-приложения, содержащего упражнения для закрепления изученного материала.

Для достижения этой цели необходимо решить следующие задачи:

- провести анализ предметной области и изучить методические материалы преподавателя английского языка ВКИ НГУ М.В. Караваевцевой;
- определить требования к функциональным возможностям веб-приложения;
- выбрать программные среды и средства разработки;
- разработать алгоритмы реализации основных функций веб-приложения;
- создать пользовательский интерфейс, соответствующий стилю сайта колледжа;
- разработать серверную и клиентскую части веб-приложения;
- провести тестирование программного средства;
- опубликовать приложение на веб-сервере.

В процессе разработки использовались языки разметки HTML и CSS, язык программирования JavaScript вместе с библиотекой React, а в качестве среды разработки – Visual Studio Code.

Практическая значимость данной выпускной квалификационной работы заключается в создании специализированного программного средства, которое будет помогать студентам в выполнении практических упражнений по английскому языку.

1 ПОСТАНОВКА ЗАДАЧИ ВКР

Для разработки программного средства в рамках выпускной квалификационной работы необходимо четко определить и сформулировать задачи, которые включают в себя бизнес-требования, пользовательские и системные требования, а также требования к графическому интерфейсу и плану выполнения ВКР.

1.1 Бизнес-требования

Разрабатываемое программное средство предназначено для изучения английского языка, с акцентом на потребности студентов Высшего колледжа информатики. Основная цель данного проекта – предоставить студентам удобный и эффективный инструмент для изучения и закрепления английского языка на начальном уровне.

Данное веб-приложение будет включать в себя разнообразные упражнения и теоретический материал, основанный на учебном пособии преподавателя английского языка ВКИ НГУ М.В. Караваевцевой.

Основные бизнес-цели:

1. Возможности выполнения упражнений в электронной форме:

Программное средство будет предлагать разнообразные задания для практики навыков чтения, письма, аудирования и разговорной речи, представленных в удобном для выполнения веб-формате.

2. Автоматизация процесса самоконтроля знаний:

Система будет обеспечивать возможность проверки выполненных упражнений.

3. Повышение мотивации студентов к изучению английского языка:

Интерактивность и доступность учебных материалов через веб-приложение будут способствовать увлечению и заинтересованности студентов в изучении английского языка.

Разработанное веб-приложение будет полезно как для работы в аудитории, так и для самостоятельного обучения дома. В учебных аудиториях преподаватели смогут использовать его для проведения занятий, предоставления студентам

дополнительных упражнений и контроля их выполнения. Вне аудитории студенты смогут продолжать обучение, выполнять домашние задания и самостоятельно повторять пройденный материал, что сделает процесс изучения более непрерывным и эффективным.

1.2 Пользовательские требования

Разрабатываемое программное средство должно предоставлять пользователям широкий спектр возможностей, обеспечивающих всестороннее и эффективное изучение английского языка. Ниже приведены основные функции и особенности, которые должны быть реализованы для достижения этой цели:

1. Изучение грамматики английского языка

Программное средство должно включать разделы, посвящённые различным аспектам грамматики:

- Времена глаголов: настоящее, прошедшее и будущее время с их аспектами (простое, продолженное, совершенное и т.д.), включающие объяснения и упражнения;
- Артикли: правила использования определённых и неопределённых артиклей (a, an, the) в различных контекстах, с примерами и практическими заданиями;
- Модальные глаголы: изучение значений и использования глаголов, таких как can, could, may, might, must и т.д.

2. Развитие и пополнение словарного запаса

Веб-приложение должно способствовать расширению словарного запаса через:

- Тематические списки слов;
- Контекстные упражнения: задания для использования новых слов в предложениях и диалогах.

3. Выполнение упражнений для закрепления грамматики и лексики

Эффективное обучение обеспечивается через:

- Грамматические упражнения: задания для проверки и закрепления грамматических знаний;
- Лексические упражнения: упражнения для практики новой лексики,

такие как подбор синонимов и использование слов в контексте;

- Комбинированные упражнения: задания, интегрирующие грамматику и лексику.

4. Представление справочной информации в виде таблиц

Для удобства студентов теоретическая информация должна быть представлена в наглядной форме:

- Грамматические таблицы: отображающие правила использования времен и другие грамматические аспекты;
- Лексические таблицы: тематические списки слов, таблицы синонимов и антонимов;
- Дополнительные материалы: правила произношения, примеры разговорных фраз и идиом.

5. Использование интерактивных методов обучения

Программное средство должно использовать интерактивные методы для повышения эффективности обучения, например интерактивные задания.

1.3 Системные требования

Программное обеспечение должно состоять из следующих основных модулей, каждый из которых играет ключевую роль в обеспечении функциональности и удобства использования веб-приложения:

- Модуль справочных материалов, предоставляющий сжатую и структурированную информацию о грамматике английского языка, соответствующую рабочей программе ВКИ НГУ для студентов первого курса;
- Модуль практических упражнений, предназначенный для закрепления знаний по грамматике и увеличения словарного запаса. По результату выполнения предоставляется отчет о количестве верных ответов.
- Модуль графического интерфейса, который отвечает за интуитивно понятный пользователям интерфейс, имеет адаптивный дизайн для смартфонов и соответствует дизайну сайта ВКИ.

Эти системные требования создают эффективную среду для изучения и закрепления материала, обеспечивая удобство и доступность для студентов.

1.4 Требования к графическому пользовательскому интерфейсу

Требования к внешнему виду пользовательского интерфейса включают в себя несколько ключевых аспектов, направленных на обеспечение простоты, удобства и эстетической привлекательности.

Интерфейс должен быть простым и интуитивно понятным, чтобы пользователи могли легко ориентироваться и получать доступ ко всем функциям программного обеспечения. Простота в дизайне обеспечит минимальные затраты времени на обучение и адаптацию к приложению. Цветовая схема интерфейса должна совпадать с дизайном сайта образовательной организации, создавая единый визуальный стиль и повышая узнаваемость. Это способствует не только эстетической гармонии, но и укреплению бренда образовательного учреждения.

Шрифты, используемые в интерфейсе, должны быть четкими и хорошо читаемыми на различных устройствах, включая ПК, планшеты и смартфоны. Это обеспечит комфортное восприятие информации и снизит утомляемость глаз при длительном использовании приложения. Иконки и кнопки должны быть функционально очевидными и достаточно большими для удобства использования на сенсорных экранах. Это особенно важно для студентов, которые будут использовать приложение на мобильных устройствах, так как крупные и понятные элементы интерфейса снижают вероятность ошибок и улучшают пользовательский опыт.

Адаптивный дизайн интерфейса должен обеспечивать корректное отображение на различных устройствах. Независимо от того, используется ли приложение на компьютере, планшете или смартфоне, все элементы интерфейса должны автоматически подстраиваться под размер экрана, сохраняя свою функциональность и удобство.

Требования по доступу к функциональности системы через пользовательский интерфейс также играют важную роль в обеспечении эффективности и удобства использования приложения.

Пользователи должны иметь возможность легко переходить между различными модулями приложения, будь то справочные материалы, практические упражнения или настройки. Это позволит студентам быстро находить нужные разделы и экономить время. Важно обеспечить быстрый доступ к текущим

заданиям или материалам из любой точки приложения, чтобы студенты могли продолжать обучение без необходимости многоократного перехода между разделами.

Кроме того, пользователи должны иметь возможность изменять настройки программы в соответствии с их личными предпочтениями. Это включает в себя возможность регулировки размера шрифта и изменения цветовой схемы, что особенно важно для пользователей с особыми потребностями в области зрения.

Индикатор, показывающий правильные и неправильные ответы, должен быть видимым и четким, обеспечивая мгновенную обратную связь и помогая студентам отслеживать свой прогресс и исправлять ошибки.

1.5 План-график выполнения ВКР

Составленный план-график помогает правильно планировать время выполнения выпускной квалификационной работы. В таблице 1 подробно представлены этапы разработки ПС.

Таблица 1 – План-график ВКР

№ п/п, наименование	Наименование работ	Срок (дни)	Даты
Формирование технического задания	Составление функциональных и технических требований	12	01.12 – 12.12
Прототипирование клиентской части	Составление прототипа дизайна сайта	7	13.12 – 20.12
Прототипирование серверной части (сервер)	Создание прототипа серверного взаимодействия с клиентом	10	21.12 – 31.12
Разработка клиентской части	Разработка дизайна сайта	12	10.01 – 22.01
Разработка серверной части (сервер)	Разработка серверного взаимодействия с клиентом	15	23.01 – 08.02
Наполнение контентом	Упражнения, справочные материалы	20	09.02 – 29.02
Тестирование	Проверка корректности	10	01.03 – 10.03

взаимодействия	отображения элементов интерфейса, а также переходов между ними. Проверка работоспособности.		
Исправление ошибок	Исправление ошибок в порядке критичности	21	11.03 – 01.04
Публикация	Публикация проекта на сервере	7	02.04 – 09.04

По данным из таблицы, для разработки потребовалось 114 дней: 12 дней на формирование технического задания, 17 дней на прототипирование, 47 дней на разработку, 31 день на тестирование и исправление ошибок, и 7 дней на публикацию на сервере.

Детальное представление план-графика ВКР позволяет быстро реагировать на выявленные проблемы и изменения в процессе разработки, выполняя работу в срок и сохраняя качество программного продукта.

2 АНАЛИЗ ТРЕБОВАНИЙ И ОПРЕДЕЛЕНИЕ СПЕЦИФИКАЦИЙ

2.1 Описание предметной области задачи ВКР

Разработка программного обеспечения для изучения английского языка требует глубокого понимания предметной области, включая ключевые информационные объекты и взаимосвязи между ними. Это позволяет создать функционально обоснованное и удобное приложение, отвечающее потребностям пользователей.

2.1.1 Информационные объекты предметной области и взаимосвязи между ними

Информационные объекты в предметной области «Разработка программного обеспечения для изучения английского языка» можно разделить на два типа:

1. Учебные материалы:

- Тексты для чтения на английском языке: обеспечивают студентов материалом для практики чтения и понимания английского языка, помогая улучшать навыки восприятия письменного текста;
- Грамматические правила и конструкции: содержат теоретическую информацию о грамматике английского языка, необходимую для правильного построения предложений;
- Слова и выражения: включают тематические списки слов и выражений, способствующие расширению словарного запаса студентов;
- Упражнения и задания для закрепления материала: практические задания, направленные на закрепление изученного материала и проверку знаний студентов.

2. Профиль пользователя:

- Персональные данные: информация о пользователе, такая как имя, контактные данные и учебная группа;
- История обучения: запись прогресса пользователя, включая выполненные задания и изученные темы;

- Статистика и показатели успеваемости: данные о результатах выполнения упражнений, оценки и другие показатели, отражающие успехи студента;
- Индивидуальные настройки и предпочтения: персонализированные настройки интерфейса и предпочтения пользователя, такие как размер шрифта, цветовая схема и прочее.

2.1.2 Информационные и функциональные потребности пользователей разрабатываемой ПС

Основные информационные и функциональные потребности пользователей разрабатываемого программного обеспечения для изучения английского языка включают в себя следующее:

1. Информационные потребности пользователей:

Доступ к грамматическим правилам английского языка с примерами и пояснениями: студенты нуждаются в доступе к подробным грамматическим правилам с иллюстрациями и разъяснениями, чтобы правильно понимать основы языка.

Возможность изучения правил чтения и транскрипции: пользователи ожидают возможности изучать правила чтения английского языка и его транскрипцию для более эффективного освоения произношения.

2. Функциональные потребности пользователей:

Выполнение интерактивных упражнений для закрепления грамматики и лексики: студенты желают выполнять разнообразные упражнения, которые помогут им закрепить изученный материал и улучшить навыки в грамматике и лексике.

Использование справочников в процессе работы с учебными материалами: возможность обращения к справочным материалам во время выполнения упражнений поможет студентам быстро разрешать возникающие вопросы и повысит эффективность обучения.

2.1.3 Методы работы с информационными объектами предметной области

При использовании веб-приложения "IT-English" студенты Высшего колледжа информатики Новосибирского государственного университета (ВКИ НГУ), изучающие английский язык, вступают во взаимодействие с различными информационными объектами, представленными в приложении. Этот процесс организован таким образом, чтобы обеспечить эффективное усвоение материала и максимальное удобство для пользователей конкретной целевой аудитории.

При переходе по ссылке и загрузке веб-приложения пользователи, являющиеся студентами ВКИ НГУ и изучающими английский язык, получают доступ к разнообразным возможностям обучения. Они могут выбрать направление работы, открыть справочный материал или приступить к выполнению практических упражнений.

Справочный материал представляет собой дополнительный источник теоретической информации, который помогает студентам углубить свои знания. После открытия справочного материала студенты имеют возможность ознакомиться с теoriей, изучить различные грамматические конструкции, а затем закрыть его для продолжения работы.

Выполняя практические упражнения, студенты закрепляют изученный материал и улучшают свои навыки. Важным аспектом работы с упражнениями является получение обратной связи о правильности или неправильности ответов. Веб-приложение автоматически анализирует ответы студентов и предоставляет информацию о количестве правильных и неправильных ответов по завершению упражнения. Это позволяет студентам оценить свой прогресс, выявить пробелы в знаниях и сфокусироваться на повторении теоретического материала для дальнейшего совершенствования.

2.1.4 Обзор существующих программных реализаций решения задачи

В настоящее время существует несколько популярных приложений и онлайн-платформ для изучения английского языка, каждое из которых обладает своим уникальным функционалом и особенностями.

Одним из наиболее известных является Duolingo – мобильное приложение и

веб-сервис, пользующиеся широкой популярностью благодаря своему бесплатному базовому функционалу. Скриншот данного веб-сервиса отображен на рисунке 1.

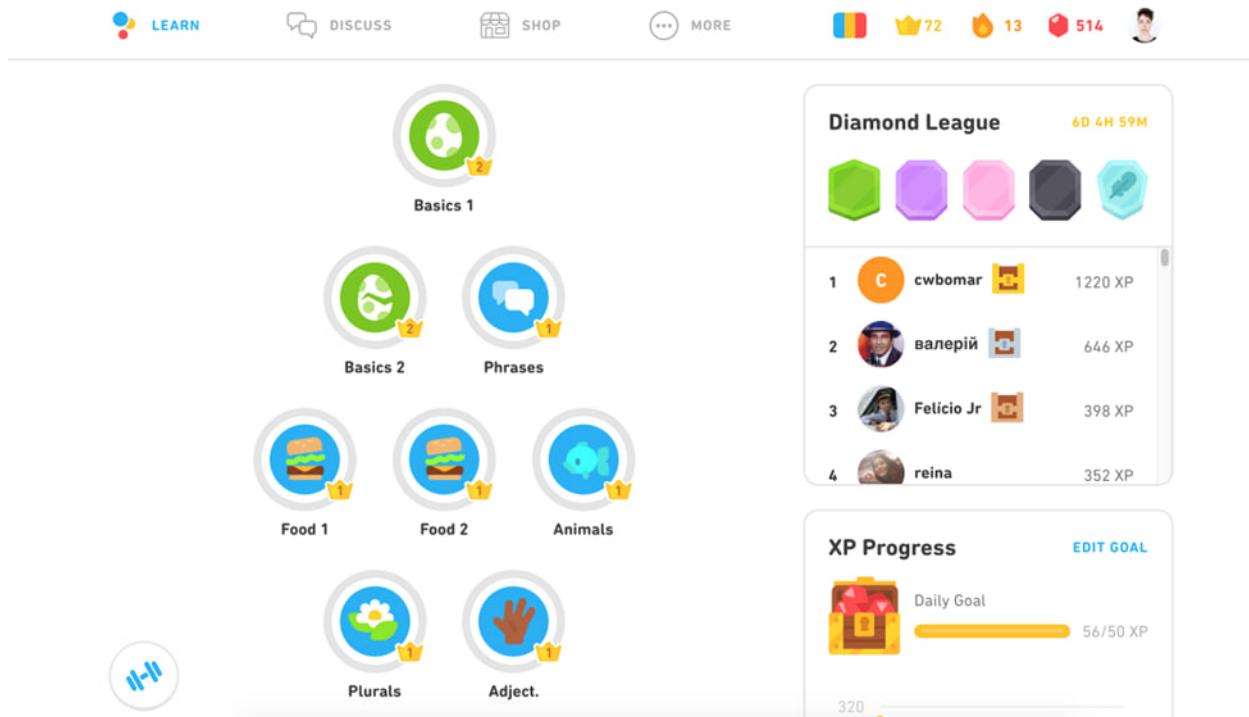


Рисунок 1 – Скриншот веб-сервиса «Duolingo»

Преимущества Duolingo включают:

- Удобный интерфейс, который обеспечивает простоту в использовании.
- Разнообразие упражнений, включая чтение, письмо, аудирование и разговорную практику.
- Возможность создания личного словаря для сохранения новых слов и выражений.

Однако у Duolingo также есть недостатки:

- Ограниченный объем доступного материала в бесплатной версии, что может ограничить пользователей в доступе к полноценному обучению.
- Отсутствие полноценных объяснений грамматики, что может затруднить понимание основных правил и конструкций языка.

Следующим рассмотренным аналогом является Lingualeo – онлайн-платформа, предоставляющая обучение в игровой форме и возможность изучения нескольких языков. Этот сервис представлен на рисунке 2.

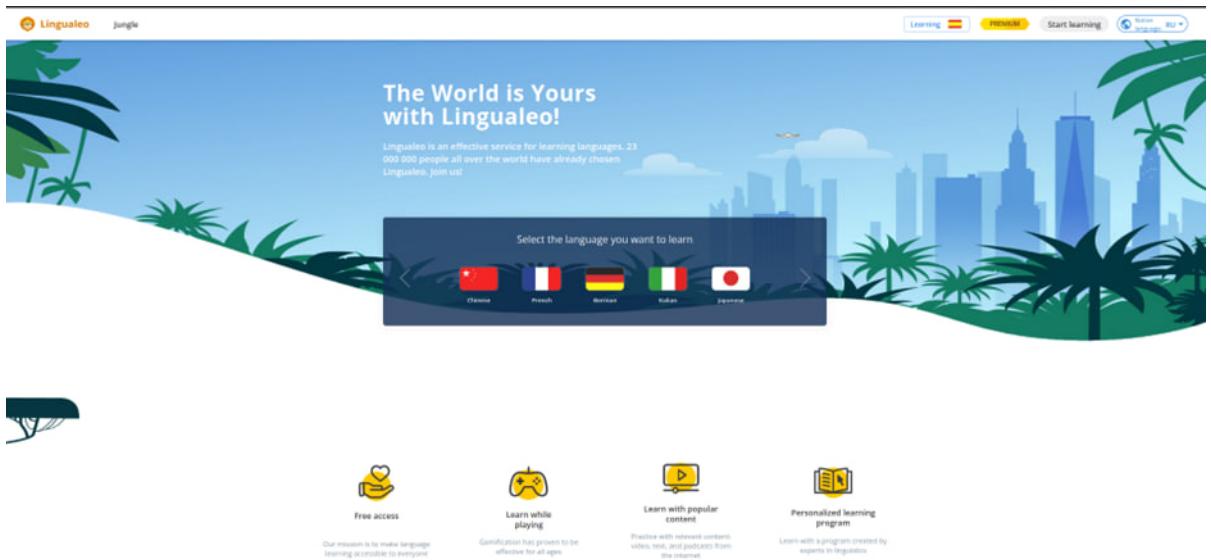


Рисунок 2 – Скриншот веб-сервиса «Lingualeo»

Плюсы Lingualeo включают наличие мотивирующих игровых элементов, которые помогают пользователям удерживать интерес к обучению.

Однако у Lingualeo имеются минусы:

- Неудобный интерфейс, который может затруднять взаимодействие пользователя с платформой.
- Высокая цена подписки, что делает этот сервис менее доступным для некоторых пользователей.
- Отсутствие обучения английскому языку, что является значительным недостатком для студентов и всех, кто интересуется изучением этого языка.

Еще одной из анализируемых платформ является Rosetta Stone – уникальная система обучения, которая предлагает методику обучения без перевода на родной язык. Данная платформа представлена на рисунке 3.

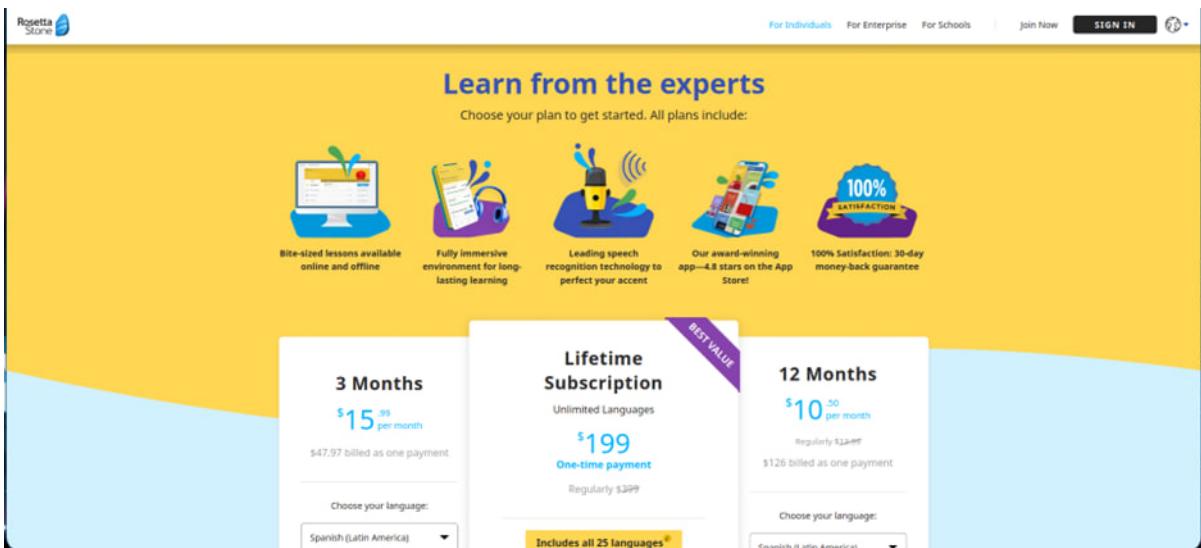


Рисунок 3 – Скриншот веб-сервиса «Rosetta Stone»

Среди преимуществ Rosetta Stone следует отметить:

- Эффективность для начинающих, благодаря уникальной методике, которая позволяет погружаться в языковую среду без использования родного языка.
- Качественные обучающие материалы, которые способствуют глубокому пониманию и закреплению изучаемого материала.

Однако у Rosetta Stone имеются недостатки:

- Высокая цена, что может стать препятствием для некоторых пользователей.
- Ограниченный набор грамматических тем и лексики, что ограничивает возможности для расширения знаний.
- Отсутствие обучения Русско-Английского языка, что может быть значительным недостатком для российских студентов, стремящихся улучшить свои навыки в переводе и владении английским.

Проведя анализ существующих решений, можно сделать вывод, что все они являются платными и предлагают обобщенный тип знаний, ориентированный на широкую аудиторию. Однако в контексте разработки веб-приложения "IT-English" предполагается сделать упор на предоставление доступных упражнений и материалов, специально адаптированных для студентов ВКИ НГУ. Подход к обучению будет основан на учебных материалах и методиках, преподаваемых в колледже, что позволит студентам более эффективно закреплять изученный

материал и успешно справляться с учебными заданиями.

2.1.5 Концептуальное обоснование разработки

Проведенный анализ предметной области и обзор существующих решений позволяют сделать вывод о наличии определенных недостатков и ограничений в существующих программах и приложениях для изучения английского языка. Это включает в себя, в частности, ограниченный функционал, высокую стоимость использования, а также несовершенство в адаптации материалов под конкретные образовательные потребности.

На основе этого вывода становится очевидной необходимость разработки специализированного программного обеспечения, которое могло бы эффективно устранить указанные недостатки и ограничения.

Отличительной особенностью разрабатываемой программы будет использование материалов, применяемых в обучении в ВКИ НГУ. Это подход позволит не только обогатить обучающий контент более актуальными и качественными материалами, но и значительно упростить процесс подготовки студентов к экзаменам, поскольку они будут работать с материалами, знакомыми им из учебного процесса.

2.2 Классы и характеристики пользователей

При анализе классов и характеристик пользователей разрабатываемого программного обеспечения для изучения английского языка выделяются несколько ключевых групп пользователей: ученик, преподаватель и администратор.

1. Учащиеся:

- Социальные характеристики: студенты Высшего колледжа информатики Новосибирского государственного университета (ВКИ НГУ);
- Мотивационная среда: стремление к повышению навыков владения английским языком для успешного обучения и будущей профессиональной деятельности в сфере информационных технологий;
- Навыки и умения: базовые навыки работы с компьютером и

интернетом;

- Требования к ПО: веб-браузер (например, Chrome, Firefox), доступ к интернету;
- Задачи пользователя: изучение грамматики, расширение словарного запаса, выполнение упражнений для закрепления материала, подготовка к экзаменам;
- Рабочая среда: персональный компьютер или ноутбук, доступ к интернету в учебных аудиториях или дома.

2. Преподаватель:

- Социальные характеристики: специалисты в области преподавания английского языка в Высшем колледже информатики Новосибирского государственного университета (ВКИ НГУ);
- Мотивационная среда: стремление к эффективному обучению студентов, использование современных образовательных технологий;
- Навыки и умения: высокий уровень владения английским языком, умение работать с образовательными программами и технологиями;
- Требования к ПО: веб-браузер, доступ к интернету, возможность работы с учебными материалами;
- Задачи пользователя: подготовка и проведение занятий, создание учебных материалов, мониторинг прогресса студентов, адаптация учебного процесса под потребности группы;
- Рабочая среда: персональный компьютер или ноутбук, доступ к интернету в учебных аудиториях или дома.

3. Администратор:

- Социальные характеристики: специалисты по обслуживанию и поддержке веб-приложения в Высшем колледже информатики Новосибирского государственного университета (ВКИ НГУ);
- Мотивационная среда: стремление к обеспечению стабильной работы веб-приложения, удовлетворение потребностей пользователей;
- Навыки и умения: умение управлять веб-хостингом, обновлять и добавлять новый учебный материал, обеспечивать безопасность приложения;

- Требования к ПО: доступ к системным ресурсам сервера, возможность управления и администрирования веб-приложением;
- Задачи пользователя: обслуживание веб-приложения, управление контентом, обеспечение безопасности и стабильной работы программы;
- Рабочая среда: персональный компьютер или серверное оборудование, доступ к интернету.

2.3 Функциональные требования

Функциональные требования определяют поведение программной системы, которая должна быть создана для обеспечения выполнения пользовательских действий в рамках бизнес-требований и в соответствии с контекстом пользовательских запросов.

1 2.3.1 Определение функциональных возможностей ПС

В разрабатываемом веб-приложении «IT-English» выделяются следующие основные категории пользователей: студенты, преподаватели и администраторы.

Для студента доступны следующие функции:

- Изучение грамматических и лексических материалов;
- Выполнение различных интерактивных заданий;
- Получение рекомендаций по дальнейшему обучению (стоит ли повторять материал или нет);
- Отслеживание статистики своих результатов и прогресса.

На рисунке 4 также отражены основные возможности, доступные студентам.



Рисунок 4 – UML-диаграмма вариантов использования студента

У преподавателя предусмотрены такие функции как:

- Просмотр справочных пособий, связанных с грамматикой английского языка;
- Проведение тестирования знаний и оценка результатов учеников;
- Интерактивное взаимодействие с пользователями, включая предоставление обратной связи за выполненные упражнения.

На рисунке 5 отображена диаграмма вариантов использования преподавателя.

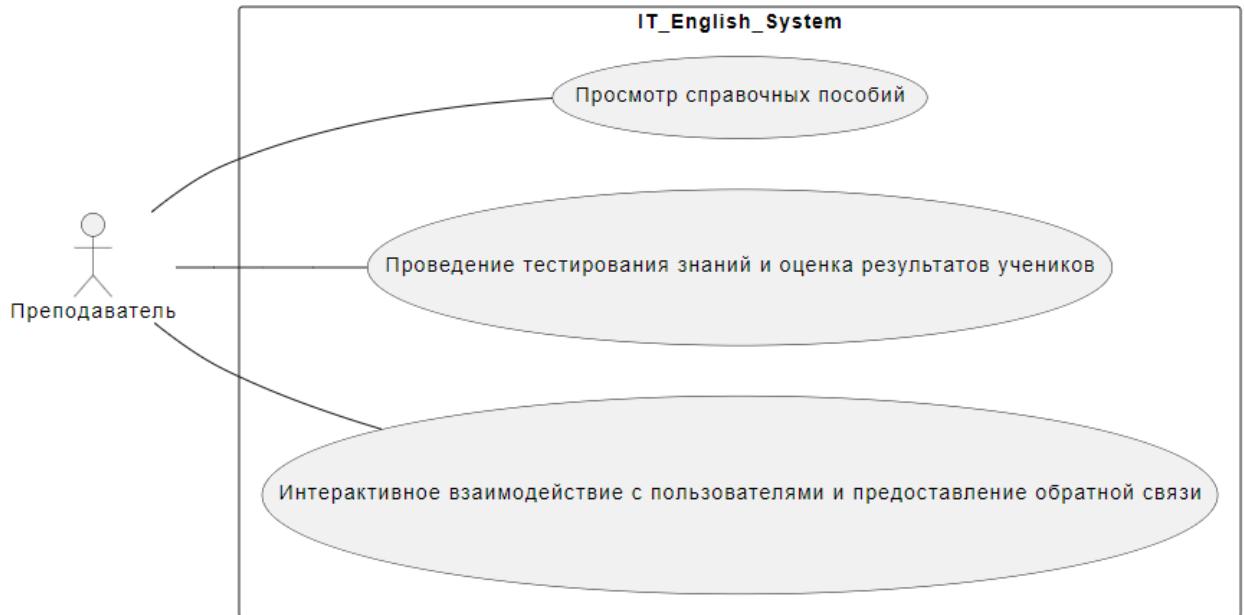


Рисунок 5 – UML-диаграмма вариантов использования преподавателя

Для администратора доступны следующие функции, которые также можно увидеть на рисунке 6:

- Добавление учебного материала в систему;
- Управление пользователями и их правами доступа;
- Мониторинг работы системы и решение технических проблем.

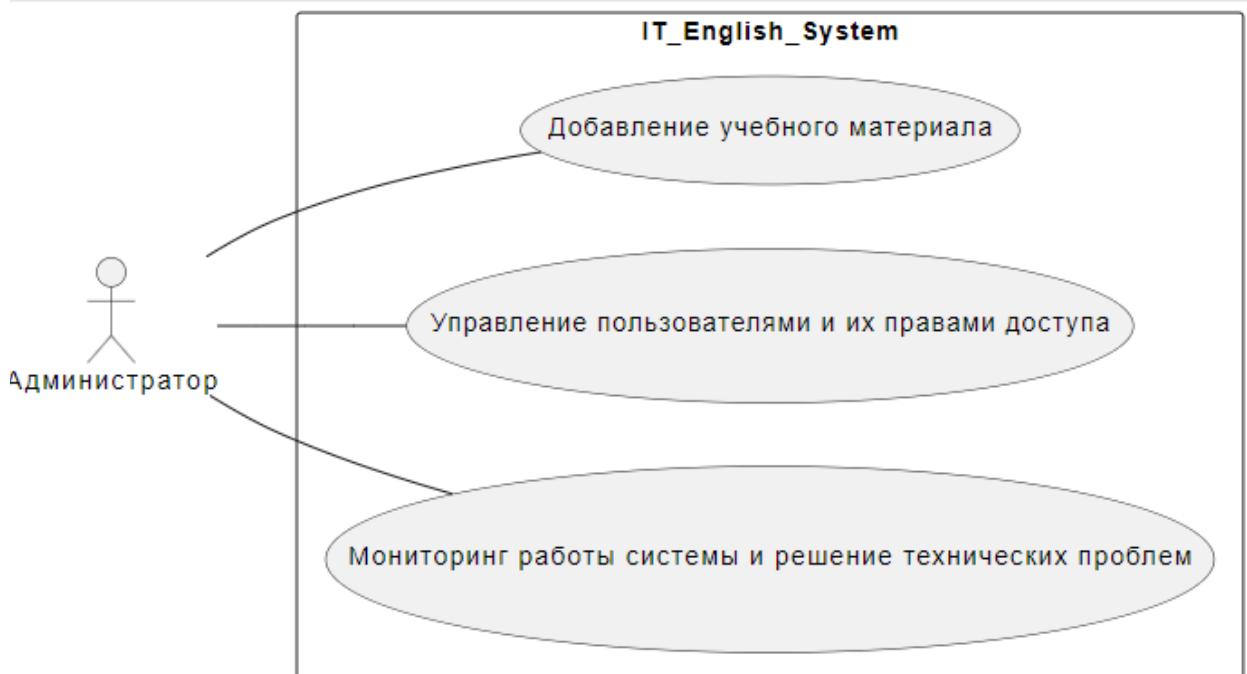


Рисунок 6 – UML-диаграмма вариантов использования администратора

2.3.2 Описание прецедентов

Каждый прецедент включает в себя определение исполнителей, целей выполнения, а также основные успешные сценарии и ссылки, обеспечивающие переход между различными частями системы. Ниже приведены таблицы, описывающие конкретные прецеденты.

Описание прецедента просмотра доступных упражнений описан в таблице 2.

Таблица 2 – Описание прецедента «Просмотр доступных упражнений»

Название прецедента	Просмотр доступных упражнений
Исполнитель	Ученик
Цель	Ознакомление с предложенными упражнениями и выбор нужного
Основной успешный сценарий	Пользователь переходит на страницу выбора упражнения, где ему предлагают список доступных занятий, которые удобно разделены на юниты учебника
Ссылки	Переход на страницу выбора, страница упражнения

Описание прецедента выполнения упражнений описан в таблице 3.

Таблица 3 – Описание прецедента «Выполнение упражнений»

Название прецедента	Выполнение упражнений
Исполнитель	Ученик
Цель	Выполнение практического упражнения
Основной успешный сценарий	Пользователь переходит на страницу упражнения. Упражнение состоит из вопросов разных типов. При ответе на один из вопросов система мгновенно оповещает пользователя о правильности каждого ответа
Ссылки	Переход со страницы выбора, страница упражнения

Описание прецедента получения результата описан в таблице 4.

Таблица 4 – Описание прецедента «Получение результата»

Название прецедента	Получения результата
Исполнитель	Ученик
Цель	Получение результата
Основной успешный сценарий	После успешного завершения всех вопросов в упражнение пользователя переводят на страницу, в которой указывается количество правильно отвеченных вопросов
Ссылки	Со страницы упражнения, страница результата

2.4 Нефункциональные требования

Нефункциональные требования представляют собой важный аспект в процессе разработки программного обеспечения. В отличие от функциональных требований, которые определяют конкретные функции системы, нефункциональные требования описывают характеристики, свойства и

ограничения, влияющие на общую производительность и качество разрабатываемой системы.

1. Производительность и масштабируемость

Программное обеспечение использует JSON-объекты для хранения данных об упражнениях. Эти объекты обеспечивают быструю передачу небольших объемов данных, но могут столкнуться с ограничениями масштабируемости. Однако, учитывая относительно невысокий объем данных, представленных заданиями из учебной программы колледжа, использование полноценной системы управления базами данных (СУБД) кажется излишним и может лишь нагрузить веб-сервер.

2. Переносимость и совместимость

Разработанное программное обеспечение работает в любом современном веб-браузере [2], что обеспечивает его доступность как на персональных компьютерах, так и на мобильных устройствах. Это позволяет студентам и преподавателям использовать программу вне зависимости от их местоположения, будь то дома или в учебном заведении.

3. Надёжность, доступность, ремонтопригодность

Большая часть вычислений обрабатывается на стороне клиента, что снижает нагрузку на веб-сервер. Основной объем нагрузки на сервер связан с загрузкой данных из базы данных, что обеспечивает стабильную и доступную работу системы. Это также облегчает обслуживание и восстановление системы в случае возникновения сбоев, повышая ее надежность и ремонтопригодность.

4. Безопасность

Поскольку веб-приложение не хранит конфиденциальные данные пользователей, риск злонамеренных атак на систему минимален. Это гарантирует высокую степень безопасности и защищенности, что особенно важно для образовательного программного обеспечения, где приоритетом является безопасное и стабильное учебное окружение.

5. Локализация

Программное обеспечение разработано исключительно для использования в рамках учебного процесса колледжа, что обеспечивает его локализацию и полное соответствие специфике образовательного контекста. Локализация программы под

конкретные нужды колледжа позволяет более эффективно использовать ее в учебном процессе и адаптировать под особенности преподавания английского языка в данном учебном заведении.

6. Удобство использования

Интерфейс программы разработан с учетом максимального удобства для пользователей и оптимизирован под мобильные устройства, обеспечивая простоту взаимодействия и интуитивную навигацию.

Система обладает высокой производительностью и надежностью, обеспечивает безопасность пользовательских данных, а также доступность и удобство использования на различных устройствах и в различных сценариях. Она точно соответствует требованиям учебного процесса колледжа и гарантирует эффективную поддержку образовательных задач студентов и преподавателей.

3 ВЫБОР ПРОГРАММНЫХ СРЕД И СРЕДСТВ РАЗРАБОТКИ

Для разработки веб-приложения «IT-English» необходимо тщательно выбрать технологический стек и средства разработки, которые обеспечат оптимальную производительность, удобство в разработке и поддержку необходимых функциональных требований. В этом разделе проведён сравнительный анализ наиболее подходящих технологических стеков, учитывая их плюсы и минусы, чтобы выбрать наилучший вариант для реализации проекта.

3.1 Сравнительный анализ имеющихся возможностей по выбору средств разработки

Для разработки требуемого программного обеспечения можно рассмотреть следующие технологические стеки:

1. React.js + Node.js [3]

Достоинства:

- Высокая производительность: React.js позволяет создавать быстрые и отзывчивые пользовательские интерфейсы, благодаря виртуальному DOM и эффективной обработке изменений.
- Простота в освоении: React.js и Node.js имеют большое и активное сообщество разработчиков, множество учебных ресурсов и примеров.
- Гибкость: возможность использовать множество библиотек и инструментов для расширения функциональности приложения.

Недостатки:

- Масштабируемость: хотя React.js и Node.js хорошо работают в небольших и средних проектах, могут возникнуть сложности с масштабированием при больших нагрузках и объемах данных.

2. PHP + Laravel [4]

Достоинства:

- Популярные и хорошо зарекомендовавшие себя технологии: PHP является одним из самых популярных языков для веб-разработки, а Laravel – мощный и удобный фреймворк для создания веб-приложений.

- Сообщество и ресурсы: большое сообщество разработчиков, множество библиотек и готовых решений, что ускоряет процесс разработки. Открытый исходный код

Недостатки:

- Меньшая производительность: по сравнению с современными стеками, такими как Node.js, PHP имеет более низкую производительность, что может быть критичным для высоконагруженных приложений.

3. Python + Django [5]

Достоинства:

- Быстрая разработка: Django – высокоуровневый фреймворк, который ускоряет процесс разработки благодаря встроенным инструментам и административному интерфейсу.
- Богатый функционал: Django предоставляет множество готовых решений для часто встречающихся задач, таких как аутентификация, управление пользователями, обработка форм и т.д.

Недостатки:

- Менее высокая производительность: по сравнению с Node.js, Python и Django могут уступать в производительности, особенно при обработке большого количества запросов в реальном времени.

Проведённый анализ показывает, что каждый из рассмотренных технологических стеков имеет свои преимущества и недостатки. Выбор конкретного стека будет зависеть от приоритетов проекта.

3.2 Характеристика выбранных программных сред и средств

Для веб-приложения «IT-English», с учётом его задач и предполагаемой нагрузки, оптимальным выбором является стек React.js + Node.js. Этот выбор обеспечит необходимую производительность и гибкость, позволяя быстро и эффективно реализовать все необходимые функциональные требования. Ниже будет проводится подробный анализ и характеристика выбранных программных сред и средств.

- React.js – современный JavaScript фреймворк, разработанный для

создания интерактивных пользовательских интерфейсов. Его отличительной особенностью является высокая производительность и эффективность. React.js позволяет разрабатывать компоненты, которые могут быть многократно использованы, что упрощает поддержку и масштабирование проекта.

- Node.js – высокопроизводительная среда выполнения JavaScript, построенная на основе движка V8 Chrome. Она позволяет создавать масштабируемые и эффективные сетевые приложения. Node.js обеспечивает асинхронную обработку запросов, что позволяет обрабатывать большое количество запросов с минимальными задержками

Данный стек выбран за счет высокой производительности, простоты в освоении, а также наличия готовых библиотек и инструментов для ускорения разработки современных веб-приложений.

4 АЛГОРИТМ РЕШЕНИЯ ПОСТАВЛЕННОЙ ЗАДАЧИ

При разработке программного продукта не менее важен алгоритм работы, который определяет последовательность этапов реализации поставленной задачи. Ниже приведены этапы реализации программной системы, отражающие ключевые шаги от формулировки требований до завершения разработки.

4.1 Этапы реализации ПС

Процесс разработки программного продукта включает в себя последовательность этапов, каждый из которых имеет определенные задачи и цели. Ниже приведены основные этапы реализации программной системы, которые также можно увидеть на рисунке 7:

- Составление функциональных и технических требований: на этом этапе определяются функциональные и технические требования к программному продукту, учитывая потребности пользователей и особенности предметной области.
- Составление прототипа дизайна сайта: разрабатывается прототип пользовательского интерфейса веб-приложения, который позволяет визуализировать основные элементы и функциональность приложения.
- Создание прототипа серверного взаимодействия с клиентом: на этом этапе создается прототип серверной части приложения, который обеспечивает основные взаимодействия с клиентской частью и обработку запросов от пользователей.
- Разработка дизайна сайта: дизайн сайта разрабатывается на основе утвержденного прототипа, учитывая требования к интерфейсу и удобство использования.
- Разработка серверной части: на данном этапе реализуется серверная часть приложения, которая обрабатывает запросы от клиентов, обеспечивает доступ к базе данных и осуществляет бизнес-логику приложения.
- Наполнение контентом: веб-приложение заполняется контентом, включающим учебные материалы, задания и другие элементы, необходимые для его функционирования.

- Тестирование взаимодействия: проводится тестирование функциональности и взаимодействия всех компонентов приложения для проверки их корректной работы.
- Исправление ошибок: выявленные на предыдущем этапе ошибки исправляются для обеспечения стабильной работы приложения и соответствия требованиям.
- Публикация проекта на сервер: после завершения разработки и тестирования приложение размещается на сервере, готовое к использованию конечными пользователями.

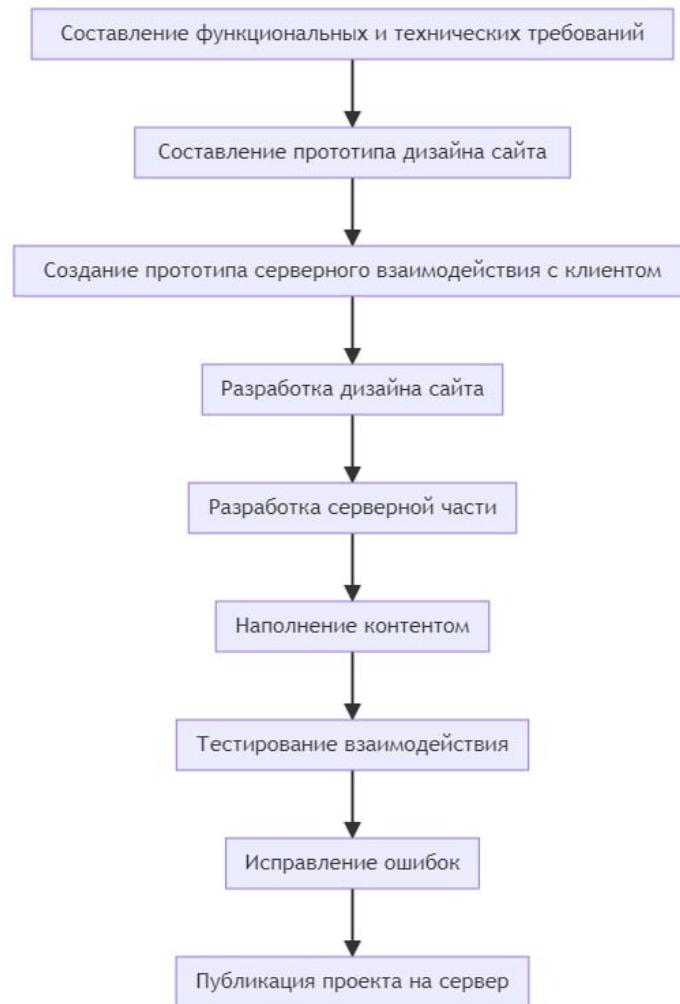


Рисунок 7 – Блок схема этапов реализации программного средства

4.2 Пользовательский интерфейс ПС

4.2.1 Взаимодействие пользователей с ПС

Чтение материалов: пользователь открывает страницу с справочными

материалами что позволяет студенту повторять теорию не имея учебника

Прохождение обучения: пользователь выбирает интересующий его юнит и упражнение. Система загружает материалы, после чего пользователь может перейти к выполнению интерактивных заданий и тестов. Система проверяет правильность ответов и выставляет результаты.

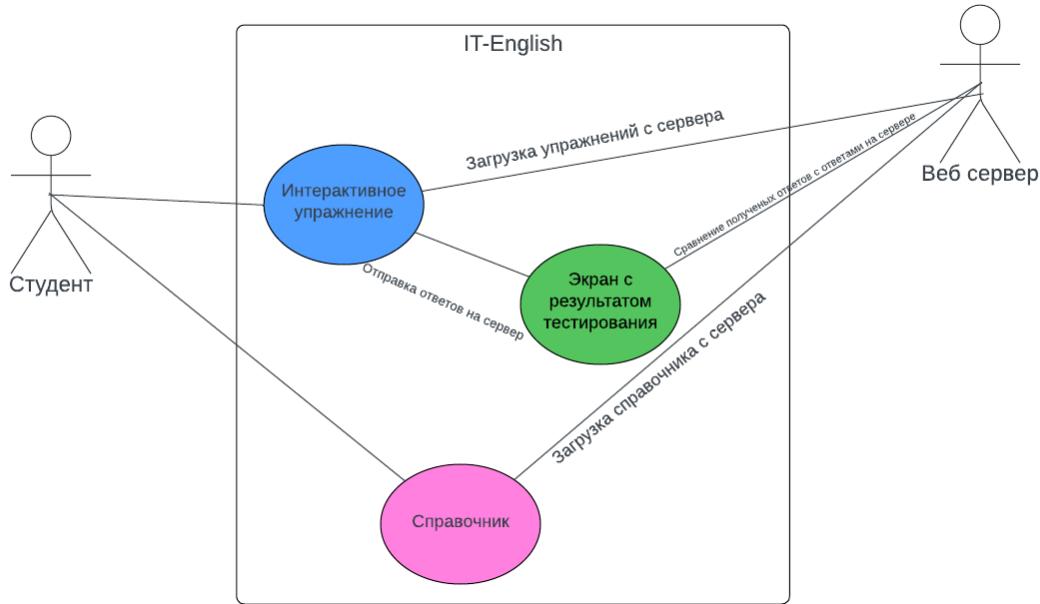


Рисунок 8. UML Диаграмма процесса обучения для пользователя

4.2.2 Проектирование пользовательских сценариев

В данном разделе представлены проекции пользовательских сценариев для каждого из основных классов пользователей веб-приложения «IT-English»:

Студент: Анна Семенова – студентка ВКИ НГУ, которая активно изучает английский язык для улучшения своих навыков в рамках учебной программы. Анна входит в систему, изучает грамматические и лексические материалы, выполняет интерактивные задания и тесты, а также отслеживает свой прогресс в изучении языка. Диаграмму последовательности данного прецедента можно увидеть на рисунке 8.

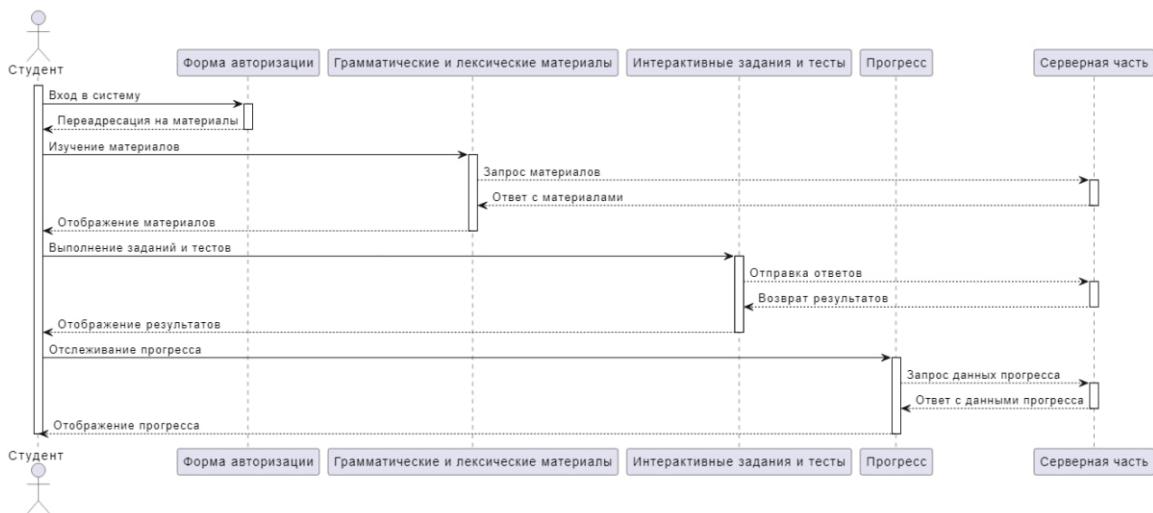


Рисунок 8 – Диаграмма последовательности для студента

Преподаватель: Алексей Коваленко – опытный преподаватель английского языка в ВКИ НГУ. Алексей использует веб-приложение для просмотра справочных материалов по грамматике английского языка, проведения тестирования знаний студентов и оценки их результатов, а также для интерактивного взаимодействия со студентами и предоставления обратной связи за выполненные упражнения. Диаграмму последовательности данного прецедента можно увидеть на рисунке 9.

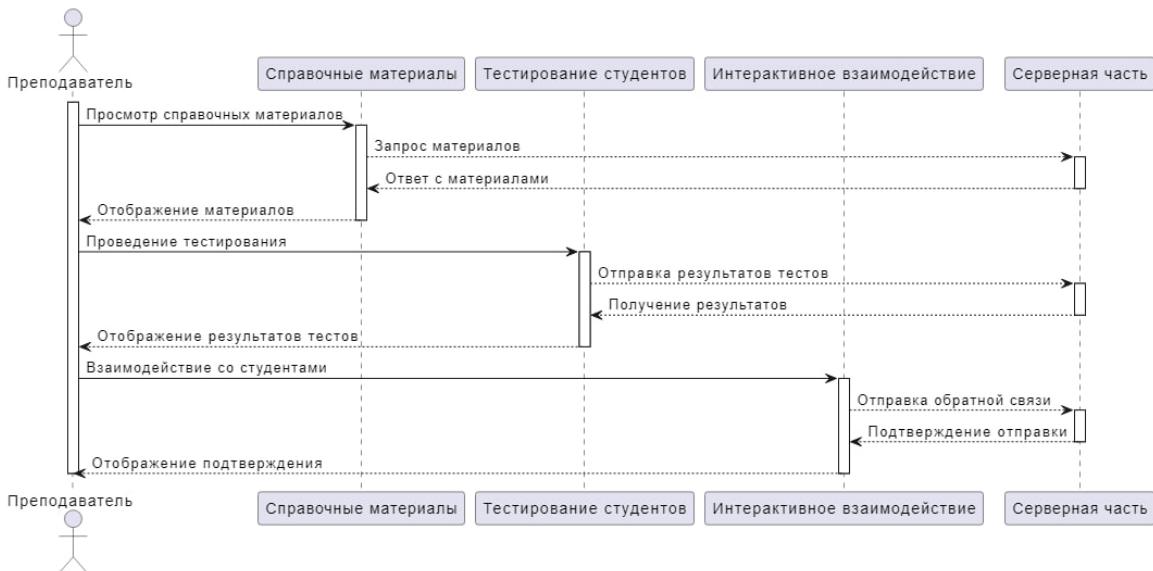


Рисунок 9 – Диаграмма последовательности для преподавателя

Администратор: Ольга Иванова – ответственная за администрирование веб-приложения «IT-English». Ольга имеет доступ к функциям добавления учебного материала в систему, управления пользователями и их правами доступа, а также

мониторинга работы системы и решения технических проблем. Диаграмму последовательности данного прецедента можно увидеть на рисунке 10.

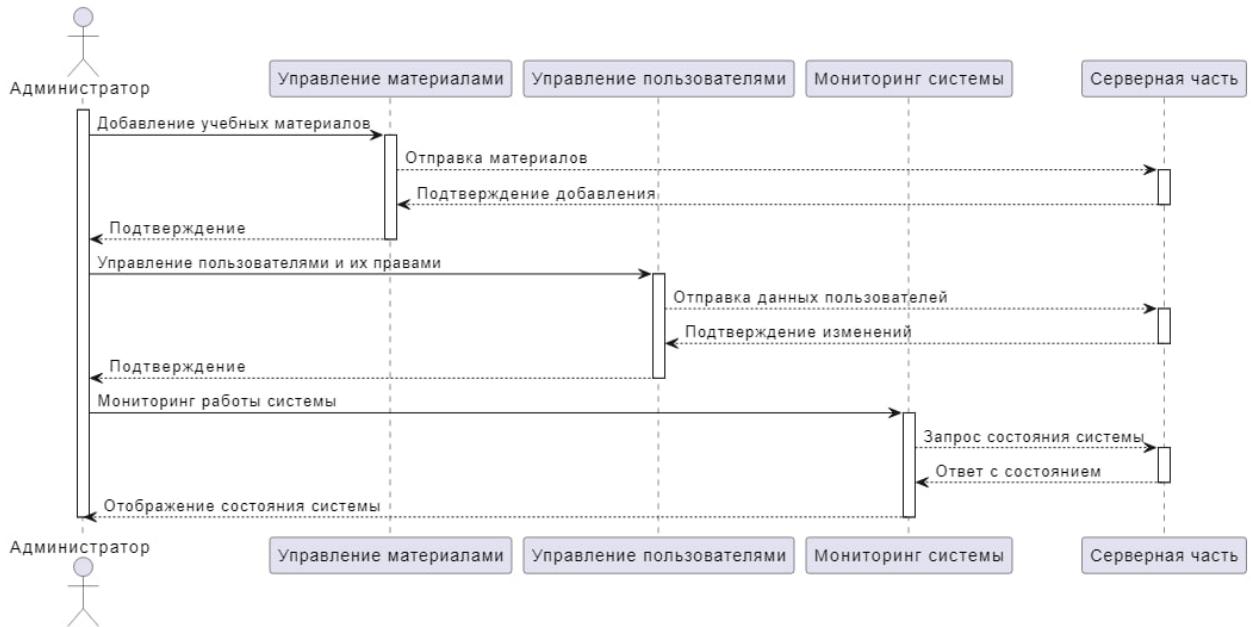


Рисунок 10 – Диаграмма последовательности для администратора

4.2.3 Определение операций пользователей

Учащиеся могут

- Выбирать и запускать интересующие курсы/уроки
- Изучать теоретические и практические материалы уроков
- Выполнять интерактивные задания и упражнения

Администратор может

- Создавать и редактировать учебные курсы и материалы

4.2.4 Составление функциональных блоков

Функциональный блок викторины

- Модули разных типов задач
- Модуль проверки ответов
- Функциональный модуль справочника
- Страницы по грамматике
- Тематические таблицы, изображения

4.2.5 Проектирование структуры экранов ПС и схемы навигации

На этапе проектирования структуры экранов веб-приложения «IT-English» формируется логическая организация интерфейса, учитывая сценарии использования и роли пользователей. Каждый экран системы представляет собой функциональное пространство, обеспечивающее определенные операции и взаимодействие пользователя. Всего было выделено шесть ключевых экранов, каждый из которых обеспечивает определенный функционал для эффективного взаимодействия пользователей с приложением.

Первым экраном является главная страница, на которой пользователи могут ознакомиться с доступными возможностями приложения и выбрать необходимую функцию. На этом экране размещены основные навигационные элементы, ведущие к различным разделам системы.

Вторым экраном является страница выбора юнита, где пользователи могут выбрать интересующий их юнит для изучения. Этот экран содержит список доступных юнитов, организованных в соответствии с учебной программой колледжа.

Третьим экраном является страница с викторинами по юниту, на которой пользователи могут выбрать конкретную викторину для прохождения. Здесь представлены все доступные викторины, связанные с выбранным юнитом, что позволяет пользователю ориентироваться в учебных материалах.

Четвертым экраном является страница викторины, где отображается содержание викторины, а пользователь может приступить к выполнению заданий. Этот экран содержит все вопросы викторины, организованные в логической последовательности.

Пятым экраном являются вопросы викторины, где пользователь отвечает на конкретные вопросы. Вопросы могут быть разных типов, а система мгновенно оповещает пользователя о правильности каждого ответа, обеспечивая интерактивное взаимодействие.

Шестым и заключительным экраном является страница с результатом, где пользователи могут просмотреть свои результаты после прохождения викторины. На этом экране отображаются оценки и рекомендации по дальнейшему обучению, что помогает пользователю определить свои сильные и слабые стороны.

Все эти экраны сформированы с учетом потребностей пользователей, обеспечивая полный цикл взаимодействия с системой «IT-English», включая аспекты выбора и выполнения учебных заданий, а также получения обратной связи. Навигационная схема, отображающая эти экраны и их взаимосвязи, представлена на рисунке 11.

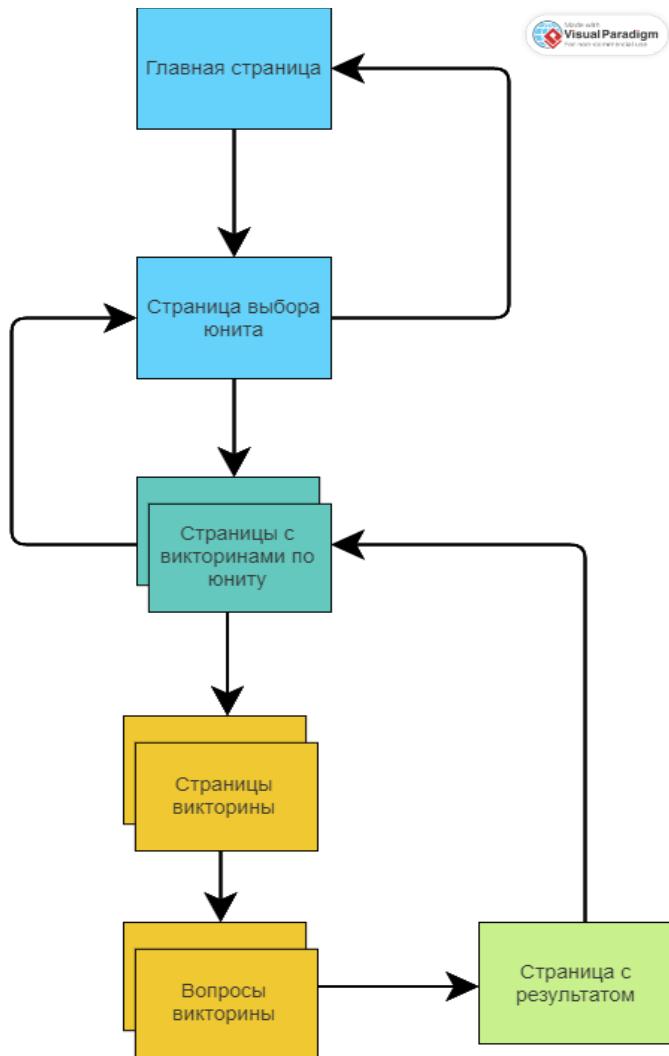


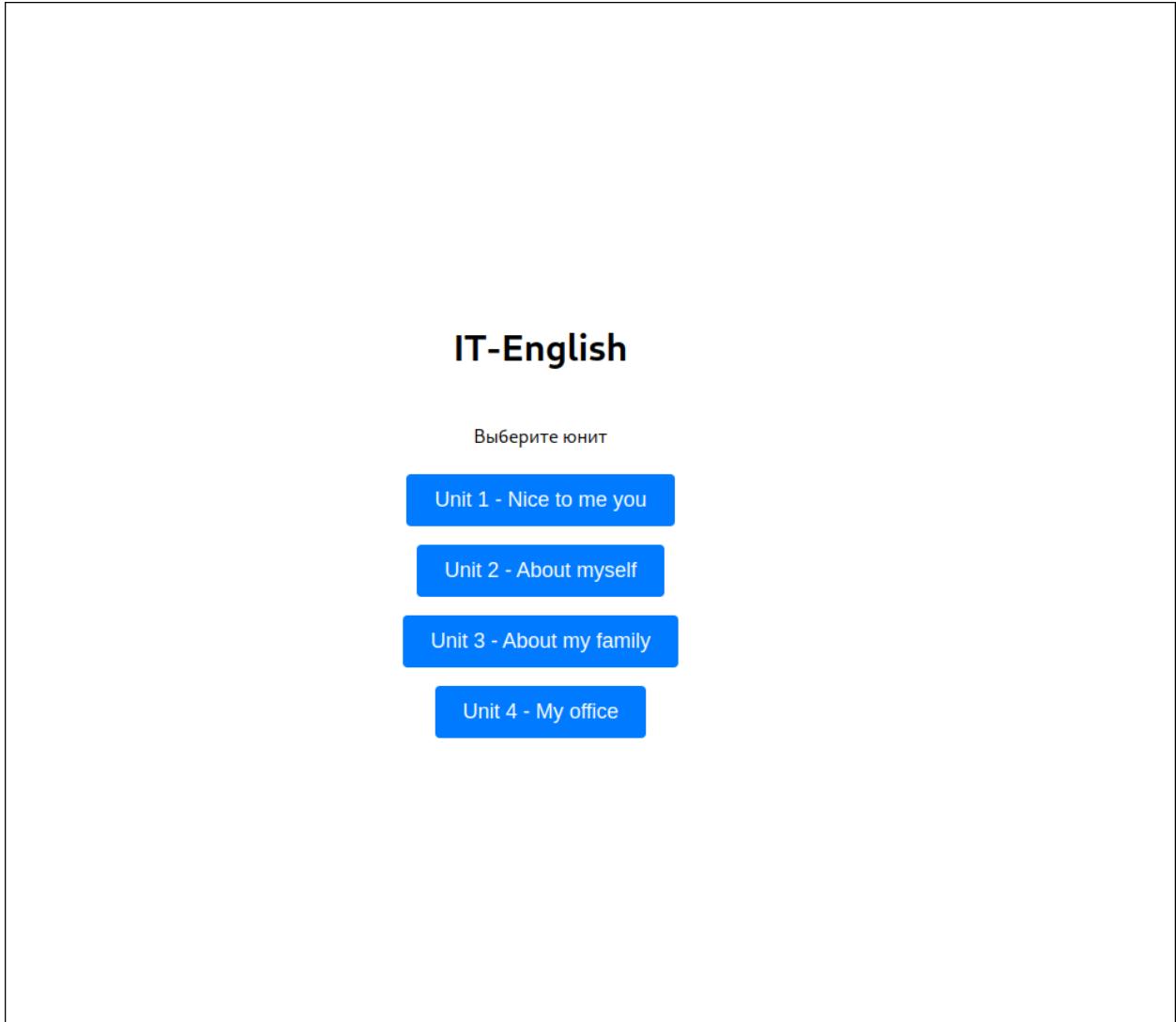
Рисунок 11 – Навигационная схема

4.2.6 Низкоуровневое проектирование

Веб-приложение состоит из множества страниц. Каждая из страниц вынесена в отдельный компонент. В списке показаны основные компоненты веб-приложения

- TitleScreen.js
- QuizSelection.js
- QuizPage.js
- Results.js

Компонент TitleScreen является главной страницой при открытии веб-приложения



и также является выбором юнита для прохождения. При выборе юнита пользователь переходит на компонент QuizSelection

Рисунок 13 — Компонент TitleScreen

Компонент QuizSelection.js отвечает за выбор упражнения в выбранном юните, Выбранное упражнение переводит пользователя на QuizPage.js

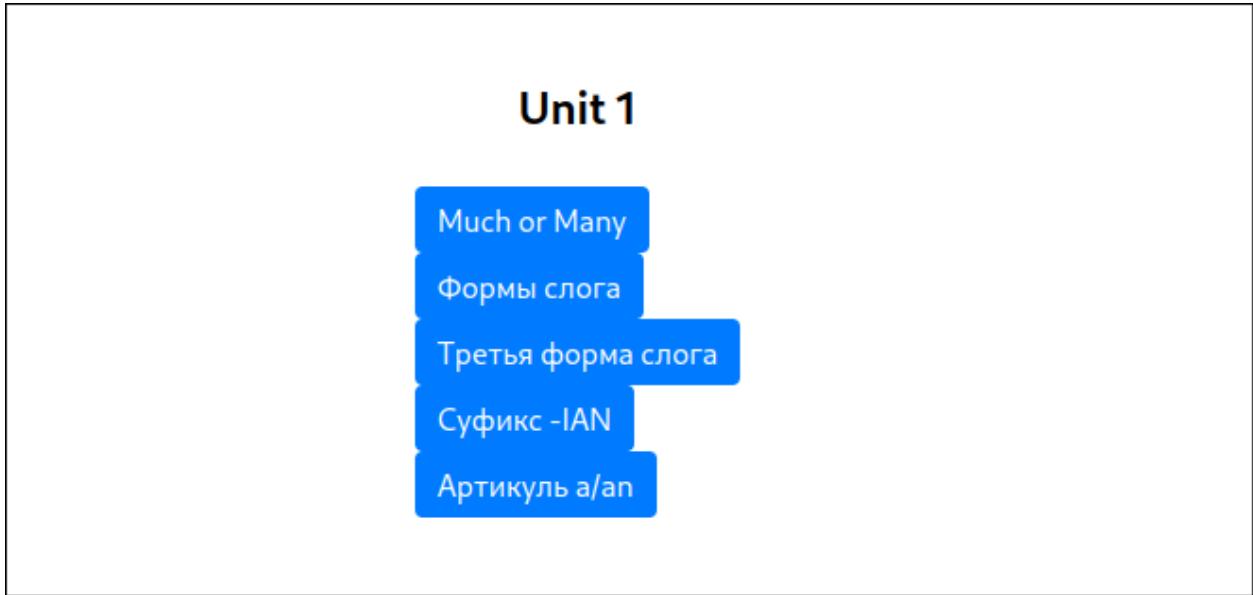


Рисунок 14 — Компонент QuizSelection.js

Компонент QuizPage отвечает за вывод страницы с упражнениями. Пользователю представляют вопросы разных типов где ему необходимо указать правильный ответ. Система реагирует на ввод пользователя и моментально указывает правильно/неправильно-ли пользователь ответил на вопрос. После ответа на все вопросы пользователя переводит на компонент Results

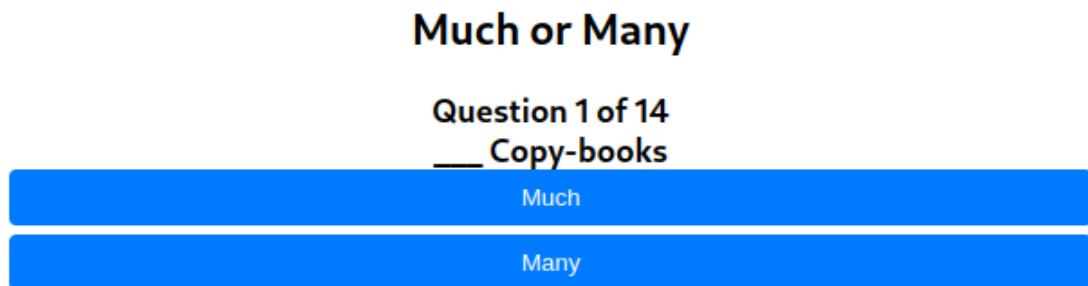


Рисунок 15 — Компонент QuizPage.js

Компонент Results выводит экран который показывает пользователю сколько правильных ответов он ввёл и предлагает вернуться на главную страницу

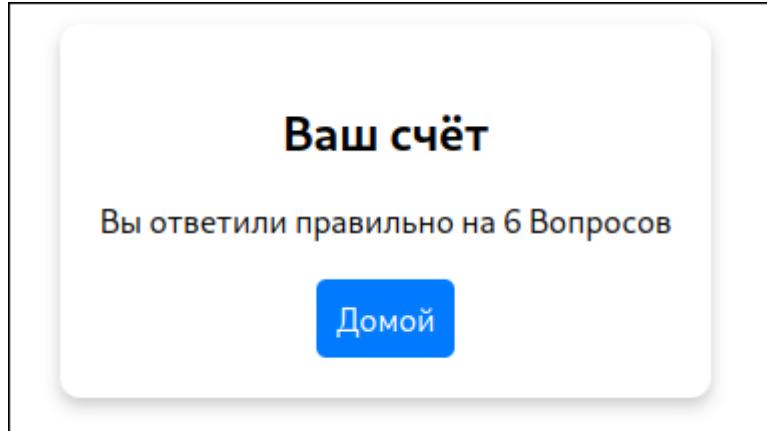


Рисунок 16 —

Компонент Results.js

4.3 Входные, выходные и промежуточные данные

Процесс взаимодействия с пользователями и обработка данных в веб-приложении «IT-English» требует четкого определения входных, промежуточных и выходных данных. Эти данные играют ключевую роль в обеспечении корректной работы системы и предоставлении пользователям необходимой информации.

1. Входные данные

- Ответы пользователя: пользователь вводит ответы на вопросы викторины, которые могут быть в виде выбранных вариантов (для вопросов с множественным выбором) или текстовых строк (для вопросов, требующих развернутого ответа). Эти данные передаются системе для последующей обработки и оценки.
- Учебные материалы: учебные материалы загружаются в систему в виде статичных страниц. Эти материалы могут включать тексты, изображения, таблицы и другие элементы, необходимые для обучения студентов. Эти страницы хранятся в системе и предоставляются пользователям по запросу.
- Упражнения: упражнения являются объектами базы данных и включают в себя информацию о заданиях, правильных ответах и рекомендациях по обучению. Эти данные используются системой для формирования викторин и тестов, которые выполняют пользователи.

2. Промежуточные данные

- Промежуточные данные включают результаты тестирования, которые сопоставляют ответы пользователя с правильными ответами. Эти данные обрабатываются системой для определения правильности ответов и формирования итоговой оценки.

3. Выходные данные

- Выходные данные формируются в виде страниц web-интерфейса, которые включают HTML для структуры страниц, CSS для оформления и JS для интерактивности. Эти страницы генерируются на основе шаблонов и данных из базы данных и предоставляются пользователям в зависимости от их действий и запросов.

Примеры таких страниц включают главную страницу, страницы выбора юнита, страницы с викторинами по юниту, страницы с вопросами викторины и страницы с результатами. Каждая из этих страниц формируется динамически, обеспечивая интерактивное и персонализированное взаимодействие пользователя с системой.

4.4 Разработка базы данных, реализуемой в рамках ПС

Упражнение (quizzes)

Хранит данные об упражнениях

Таблица 5 – Описание класса «quizzes»

Название поля	Тип поля	Описание поля
id	int	первичный ключ
name	string	Название упражнения
questions	array	Массив всех вопросов (также являются таблицами)

Вопросы с выбором(multiple-choice)

Хранит данные о вопросах

Таблица 6 – Описание класса «multiple-choice»

Название поля	Тип поля	Описание поля
type	string	Тип вопроса
questionText	string	Текст вопроса
options	array	Массив ответов
Correct answer	string	Правильный ответ

Вопросы с набиранием ответа (fill-in-the-blank)

Хранит данные о вопросах

Таблица 7 – Описание класса «fill-in-the-blank»

Название поля	Тип поля	Описание поля
type	string	Тип вопроса
questionText	string	Текст вопроса
Correct answer	string	Правильный ответ

4.6 Архитектура и схема функционирования ПС

Архитектура создаваемого программного обеспечения «IT-English» разработана с использованием модели «клиент-сервер» и базируется на современном стеке технологий JavaScript, что обеспечивает высокую производительность, масштабируемость и гибкость системы.

1. Серверная часть

Серверная составляющая приложения реализована на платформе Node.js, которая обеспечивает эффективную обработку асинхронных операций и высокую производительность. Сервер выполняет роль центрального звена системы, занимаясь обработкой и маршрутизацией запросов от клиентской части. В основе серверной логики лежит построение API (Application Programming Interface), через который происходит взаимодействие между клиентом и сервером. Это API обрабатывает поступающие запросы, осуществляет предварительную обработку и направляет необходимые ответы обратно клиентскому приложению.

Для хранения данных вместо традиционной системы управления базами данных (СУБД) были выбраны JSON-объекты. Такой подход упрощает структуру хранения данных и позволяет быстро и удобно манипулировать ними. Это также способствует гибкости расширения функциональности программного обеспечения в будущем.

2. Клиентская часть

Клиентская часть представлена многостраничным веб-приложением (МПА), реализованным с использованием популярного фреймворка React.js. React.js

позволяет создавать динамичные и интерактивные пользовательские интерфейсы, что существенно улучшает пользовательский опыт. Клиентское приложение выполняется непосредственно в браузере пользователя и обеспечивает доступ ко всем функциям обучающего веб-приложения.

3. Веб-сервер

Для обеспечения надежной и быстрой передачи данных, в архитектуре приложения используется веб-сервер Nginx. Nginx отвечает за обработку входящих HTTP-запросов, предоставление статических файлов веб-страниц, а также за ускорение времени загрузки страниц за счет использования кэширования и других оптимизаций. В результате, пользователи получают доступ к обучающему контенту максимально быстро и без задержек.

На рисунке 12 отображена архитектура веб-приложения.

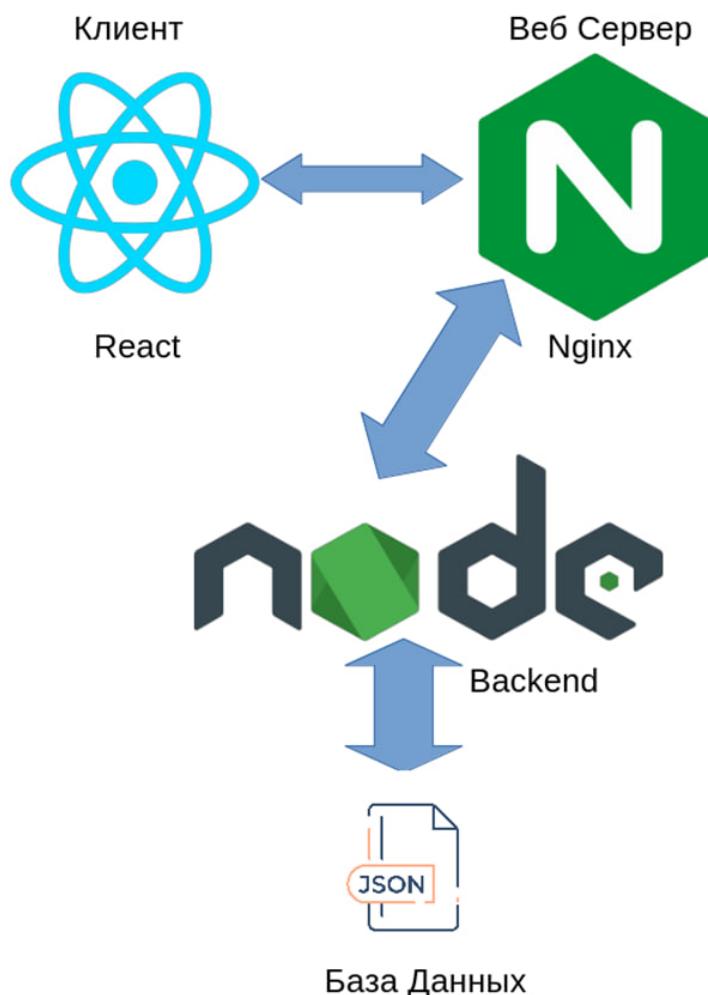


Рисунок 17 — Архитектура веб-приложения

Таким образом, архитектура и схема функционирования веб-приложения

«IT-English» демонстрируют высокую степень взаимосвязанности компонентов и использование передовых технологий, что в совокупности обеспечивает надежную, эффективную и удобную для пользователей систему обучения английскому языку в области информационных технологий.

5 ТЕСТИРОВАНИЕ И ОПТИМИЗАЦИЯ

5.1 План тестирования

Для обеспечения высокого качества и надежности разрабатываемого веб-приложения предусмотрен всесторонний план тестирования, включающий несколько ключевых направлений.

1. Функциональное тестирование

Функциональное тестирование направлено на проверку исполнения функциональных требований, предъявляемых к приложению. Оно охватывает следующие аспекты:

- Отображение и работа с учебными материалами: проверка корректности отображения текстов, видео, аудио и других учебных ресурсов, а также возможности их использования.
- Прохождение интерактивных тестов и упражнений: убедиться, что пользователи могут без проблем выполнять задания, получать обратную связь и результаты.
- Механизмы оценивания: проверка корректности работы систем оценки и автоматических экзаменов.
- Генерация аналитических отчетов: убедиться в правильной работе модулей, отвечающих за составление и представление аналитических данных о результатах пользователей.

2. Тестирование юзабилити (user experience testing)

Цель тестирования юзабилити заключается в обеспечении удобства и простоты использования веб-приложения:

- Интуитивность и понятность интерфейса: проверка, насколько легко пользователям ориентироваться в интерфейсе приложения и находить нужные функции.
- Логичность и удобство сценариев взаимодействия: анализ пользовательских сценариев и взаимодействий для обеспечения их логичности и последовательности.
- Соответствие GUI текущим трендам и стандартам: проверка, чтобы графический интерфейс пользователя соответствовал современным

стандартам и трендам в дизайне веб-приложений.

3. Нагрузочное тестирование

Нагрузочное тестирование проводится для оценки работоспособности приложения под условиями высокой нагрузки:

- Оценка масштабируемости и производительности системы: проверка системы под различными уровнями нагрузки для удостоверения ее способности выдерживать высокие объемы трафика и одновременных пользователей без потери производительности.

4. Тестирование совместимости

Тестирование совместимости предназначено для проверки работы приложения на различных платформах и устройствах:

- Тестирование на различных браузерах: убедиться, что приложение корректно работает и отображается в различных браузерах, таких как Chrome, Firefox, Edge и Safari.
- Тестирование на различных устройствах: проверка совместимости приложения на различных типах устройств, включая настольные компьютеры, ноутбуки, планшеты и смартфоны, чтобы обеспечить универсальный пользовательский опыт.

5.2 Результаты тестирования

Для тестирования использовалась утилита Apache Jmeter [7]. Утилита позволяет производить стресс тест путем отправления http запросов с нескольких потоков симулируя большой наплыв пользователя

Был произведен тест со наплыва 10 тысяч пользователей в течении 60 секунд. Тест был автоматически повторен 50 раз, результаты указаны в таблице снизу

Таблица 5 – Результаты стресс-теста

Максимальное потребление ОЗУ	Среднее время ответа (В миллисекундах)	Количество ошибок
842МВ	612	0

По данным таблицы веб-приложение использует не больше 1 гигабайта ОЗУ при

нагрузке что позволяет его деплоинг на слабых, дешёвых серверах

6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

При открытии сайта IT-English вас будет приветствовать меню с выбором юнитов.
Вам будет необходимо выбрать нужный вам юнит из предложенных



Главная страница

При Выборе необходимого вам юнита вам будут предложены все существующие упражнения для данного юнита



Unit 1

[Much or Many](#)

[Формы слога](#)

[Третья форма слога](#)

[Суффикс -IAN](#)

[Артикуль a/an](#)

Выбор упражнений

При выполнении упражнений упражнения вам придётся отвечать на вопросы различных типов. При успешном выполнении их всех вам покажут количество правильных ответов и предложат вернуться назад на главную страницу

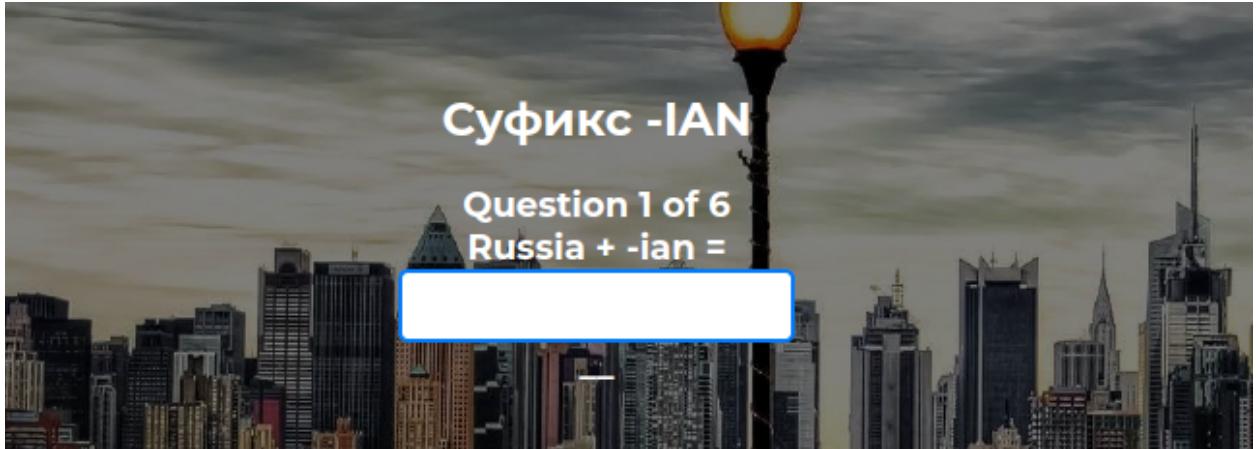
Much or Many

Question 1 of 14
Copy-books

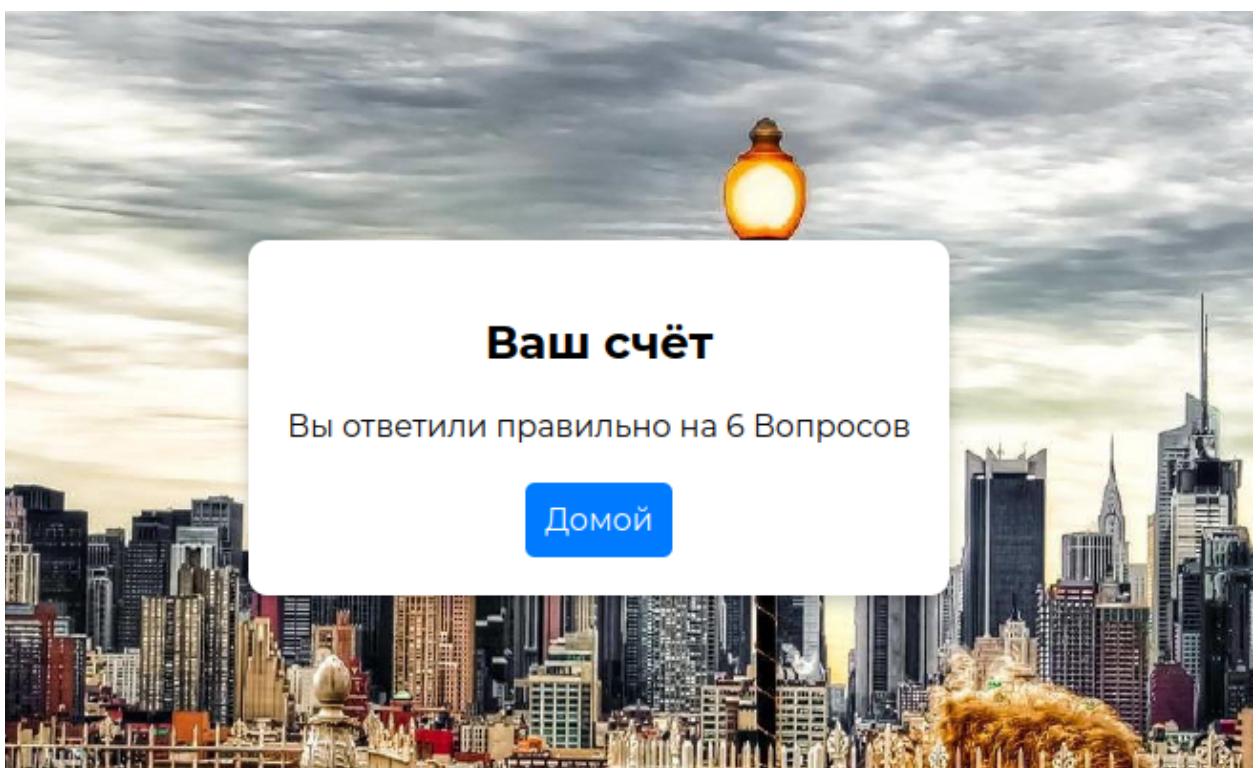
[Much](#)

[Many](#)

Выбор правильного ответа



Набор правильного ответа



Результат упражнения

ЗАКЛЮЧЕНИЕ

В рамках выпускной квалификационной работы было создано веб-приложения «IT-English». Разработка была проведена в соответствии с изначально поставленными целями и задачами, а также с учетом требований заказчика. В ходе работы над проектом были достигнуты следующие результаты:

- Был проведен анализ методических материалов преподавателя английского языка ВКИ НГУ М.В. Караваевцевой и определены требования к функциональности веб-приложения;
- Выбраны подходящие технологические стеки для реализации серверной и клиентской частей приложения;
- Разработаны эффективные алгоритмы для реализации основных функций приложения, включая управление пользователями, курсами, обучающими материалами и тестированием;
- Был разработан пользовательский интерфейс, который соответствует стилю и дизайну сайта колледжа, обеспечивая единый и целостный пользовательский опыт;
- Приложение было подвергнуто тщательному тестированию для обеспечения его стабильной работы и отсутствия ошибок. После успешного завершения тестирования приложение было опубликовано на веб-сервере и доступно для использования.

Несмотря на достигнутый успех, проект «IT-English» обладает значительным потенциалом для дальнейшего развития и улучшения. Возможные направления роста включают:

- Расширение функциональности;
- Улучшение пользовательского интерфейса;
- Добавление большее количества материалов.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

- 1 Статья Королевской академии Оксфорд “Почему английский сложно учить” <https://www.oxford-royale.com/articles/learning-english-hard/>
- 2 Список поддерживаемых браузеров у библиотеки React <https://create-react-app.dev/docs/supported-browsers-features/>
- 3 Яндекс Практикум «что такое React и как он работает»
<https://practicum.yandex.ru/blog/chto-takoe-react-i-kak-on-rabotaet/?>
- 4 SkillFactory «что такое Laravel»
<https://blog.skillfactory.ru/glossary/laravel/>
- 5 JetBrains «Состояние Django на 2024 год»
<https://blog.jetbrains.com/pycharm/2024/06/the-state-of-django/>
- 6 Пример внешнего вида сайта ВКИ НГУ <https://ci.nsu.ru/>
- 7 Набор инструментов тестирования Apache Jmeter
<https://jmeter.apache.org/>

ПРИЛОЖЕНИЯ

Приложение А

Техническое задание на разработку обучающего веб-приложения “IT-English”

1 Предмет разработки

Предметом разработки является веб-приложение для обучения студентов колледжа ВКИ НГУ базовому английскому языку.

Назначение приложения:

- предоставление и изучение обучающего материала для изучения английского языка основанных на материалах преподавателя английского языка ВКИ НГУ М.В. Карабаевой
- контроль знаний по пройденному материалу;
- расширение словарного запаса английского языка.

Цель создания приложения: создать веб-приложение для закрепления учебных материалов 1-го курса английского языка ВКИ НГУ доступного как в классе так и дома.

2 Основные функции веб-приложения

Основной функцией веб-приложения является – выполнение упражнений по английскому языку основанные на обучающей программе колледжа

3 Требования к техническому обеспечению

Устройства имеющие поддержку последних версий веб-браузеров

4 Минимальные требования к ОС

Разработанное веб-приложение должно корректно работать на всех современных операционных системах и должно корректно отображаться на экранах разных размеров

5 Требования к программному обеспечению приложения

- Для разработки веб-приложения может быть использована любая среда разработки, библиотека React, система управления версиями Git и язык программирования JavaScript
- Веб-приложение должно работать круглосуточно, без перебоев

6 Минимальное представление и структура экранов

Структура веб-приложения должна иметь следующий вид:

1. Главная страница:
2. Выбор юнитов:
 - Названия юнитов и переход между ними
3. Экран одного юнита:
 - Названия упражнений.
 - Кнопки перехода
4. Экран одного упражнения:
 - Различные типы задач
 - Отображение правильных/неправильных ответов
5. Экран результата:
 - Вывод количества правильных ответов
 - Переход на начальную страницу

7 Дизайн

Пользовательский интерфейс должен быть дружелюбным и удобным для пользователей. Все кнопки и другие интерактивные элементы должны быть интуитивно понятными без использования подсказок или инструкций.

Цветовая палитра веб-приложения должна совпадать со стилем сайта ВКИ НГУ (преимущественно белые и синие тона) [6], для выделения объектов можно использовать красные и зеленые цвета

8 Этапы работы

1. Формирование Технического задания
2. Создание прототипа
3. Разработка клиентской части
4. Разработка серверной части
5. Наполнение контентом

6. Тестирование и исправление ошибок
7. Публикация

Приложение Б

App.js

```
import React from 'react';
import { BrowserRouter as Router, Switch, Route } from 'react-router-dom';
import TitleScreen from './components/TitleScreen';
import QuizSelection from './components/QuizSelection';
import QuizPage from './components/QuizPage';
import Results from './components/Results';

const App = () => {
  return (
    <Router>
      <Switch>
        <Route path="/" exact component={TitleScreen} />
        <Route path="/select-quiz/:packNumber" component={QuizSelection} />
        <Route path="/quiz/:quizId" component={QuizPage} />
        <Route path="/results/:score" component={Results} />
      </Switch>
    </Router>
  );
};

export default App;
```

App.css

```
.App {
  text-align: center;
}

.App-logo {
  height: 40vmin;
  pointer-events: none;
}

@media (prefers-reduced-motion: no-preference) {
  .App-logo {
    animation: App-logo-spin infinite 20s linear;
  }
}
```

```
        }
    }

.App-header {
    background-color: #282c34;
    min-height: 100vh;
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    font-size: calc(10px + 2vmin);
    color: white;
}
```

```
.App-link {
    color: #61dafb;
}
```

```
@keyframes App-logo-spin {
    from {
        transform: rotate(0deg);
    }
    to {
        transform: rotate(360deg);
    }
}
```

TitleScreen.js — Главная страница

```
import React from 'react';
import { useHistory } from 'react-router-dom';
import './TitleScreen.css';

const TitleScreen = () => {
    const history = useHistory();

    const navigateToPack = (packNumber) => {
        history.push(`select-quiz/${packNumber}`);
    };

    return (
        <div className="title-screen">
            <h1>IT-English</h1>
            <p>Выберите юнит</p>
    
```

```

        <button onClick={() => navigateToPack(1)}>Unit 1 - Nice to me you </button>
        <button onClick={() => navigateToPack(2)}>Unit 2 - About myself</button>
        <button onClick={() => navigateToPack(3)}>Unit 3 - About my family</button>
        <button onClick={() => navigateToPack(4)}>Unit 4 - My office</button>
    </div>
);
};

export default TitleScreen;

```

TitleScreen.css

```

.title-screen {
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    height: 100vh;
    text-align: center;
}

.title-screen h1 {
    margin-bottom: 2rem;
}

.title-screen button {
    background-color: #007bff;
    color: #ffffff;
    border: none;
    border-radius: .25rem;
    padding: .75rem 1.5rem;
    margin: .5rem;
    font-size: large;

    transition :background-color .3s ;
}

.title-screen button:hover,
.title-screen button:focus {
    background-color:#0056b3;
    cursor:pointer;
}

```

```
}
```

TopBar.js _ навигационная панель

```
import React, { useState, useEffect } from 'react';
import './TopBar.css';
import halfMoonIcon from './half-moon-icon.svg';

const TopBar = () => {
  const [isDarkMode, setIsDarkMode] = useState(false);
  const [bgImage, setBgImage] = useState('../images/uk.jpg');
  const [fontColor, setFontColor] = useState('#000000');

  useEffect(() => {
    document.documentElement.style.setProperty('--font-color', fontColor);
  }, [fontColor]);

  const toggleDarkMode = () => {
    setIsDarkMode(!isDarkMode);
    setBgImage(isDarkMode ? '/path/to/image1.jpg' : '/path/to/image2.jpg');
    setFontColor(isDarkMode ? '#000000' : '#ffffff');
  };

  return (
    <div
      className="top-bar"
      style={{
        backgroundColor: '#007bff',
        backgroundImage: `url(${bgImage})`,
        color: fontColor,
      }}
    >
      <h1>IT-English</h1>
      <button className="dark-mode-toggle" onClick={toggleDarkMode}>
        <img src={halfMoonIcon} alt="Dark Mode Toggle" />
      </button>
    </div>
  );
};

export default TopBar;
```

TopBar.css

```
:root {  
    --font-color: #000000;  
    --bg-image: url('../images/us.jpg');  
}  
  
body {  
    color: var(--font-color);  
    margin: 0;  
    padding-top: 60px;  
    min-height: 100vh;  
    box-sizing: border-box;  
    background-image: var(--bg-image);  
    background-size: cover;  
    background-position: center;  
    background-repeat: no-repeat;  
}  
  
.top-bar {  
    position: fixed;  
    top: 0;  
    left: 0;  
    width: 100%;  
    height: 60px;  
    background-color: #007bff;  
    display: flex;  
    align-items: center;  
    justify-content: space-between;  
    padding: 0 20px;  
    z-index: 999;  
}  
  
.dark-mode-toggle {  
    background: none;  
    border: none;  
    cursor: pointer;  
    margin-right: 20px;  
}
```

```
.dark-mode-toggle img {  
    width: 24px;  
    height: 24px;  
}
```

QuizPage.js — страница упражнения

```
import React, { useState } from 'react';  
import { useParams, useHistory } from 'react-router-dom';  
import quizzes from './quizzesData';  
import './QuizPage.css';  
import MatchSetsQuestion from './MatchSetsQuestion';  
import CategorySortQuestion from './CategorySortQuestion';  
  
const QuizPage = () => {  
    const { quizId } = useParams();  
    const history = useHistory();  
    const [currentQuestionIndex, setCurrentQuestionIndex] = useState(0);  
    const [correctAnswersCount, setCorrectAnswersCount] = useState(0);  
    const [userInput, setUserInput] = useState("");  
    const [isAnswerSubmitted, setIsAnswerSubmitted] = useState(false);  
  
    const quiz = quizzes.find(quiz => quiz.id === parseInt(quizId));  
  
    if (!quiz) return <div>Quiz not found!</div>;  
  
    const checkAnswerAndProceed = (answer) => {  
        let isCorrect = false;  
        if (quiz.questions[currentQuestionIndex].type === "multiple-choice") {  
            isCorrect = answer === quiz.questions[currentQuestionIndex].correctAnswer;  
        } else if (quiz.questions[currentQuestionIndex].type === "fill-in-the-blank") {  
            isCorrect = answer.toLowerCase().trim() ===  
                quiz.questions[currentQuestionIndex].correctAnswer.toLowerCase().trim();  
        } else if (quiz.questions[currentQuestionIndex].type === "match-sets") {  
            isCorrect = JSON.stringify(answer) === JSON.stringify(quiz.questions[currentQuestionIndex].correctAnswer);  
        } else if (quiz.questions[currentQuestionIndex].type === "category-sort") {  
            isCorrect = JSON.stringify(answer) === JSON.stringify(quiz.questions[currentQuestionIndex].correctAnswer);  
        }  
  
        if (isCorrect) {  
            setCorrectAnswersCount(correctAnswersCount + 1);  
        }  
    }  
}
```

```

    }

setIsAnswerSubmitted(true);

if (currentQuestionIndex >= quiz.questions.length - 1) {
  history.push(`/results/${correctAnswersCount + (isCorrect ? 1 : 0)})`;
} else {
  setTimeout(() => {
    setCurrentQuestionIndex(currentQuestionIndex + 1);
    setUserInput("");
    setIsAnswerSubmitted(false);
  }, 1000); // Delay of 1 second to show answer feedback
}
};

const handleOptionSelect = (option) => { // Проверка если ответ вверный
  setUserInput(option);
  checkAnswerAndProceed(option);
};

const handleInputChange = (e) => {
  setUserInput(e.target.value);
};

// при нажатии на enter ответ отправляется
const handleKeyPressOnBlank = (e) => {
  if (e.key === 'Enter') {
    checkAnswerAndProceed(userInput);
  }
};

const handleMatchSetsAnswerSubmit = (answer) => { //проверка если вопросы на совпадения верны
  checkAnswerAndProceed(answer);
};

const handleCategorySortAnswerSubmit = (answer) => { //проверка если сортировочные вопросы верны
  checkAnswerAndProceed(answer);
};

return (
  <div className="quiz-container">
    <h2>{quiz.name}</h2>

```

```

<div className="question-text">
  <p>Question {currentQuestionIndex + 1} of {quiz.questions.length}</p>
  {quiz.questions[currentQuestionIndex].type === "multiple-choice" ?
    <p>{quiz.questions[currentQuestionIndex].questionText}</p>
    :
    (
      <span>
        {quiz.questions[currentQuestionIndex].questionText.split('__').map((part, index, arr) =>
          index < arr.length - 1 ? (<React.Fragment key={index}>{part}<input type="text" value={userInput}
          onChange={handleInputChange} onKeyPress={handleKeyPressOnBlank} className="blank-input"
          /></React.Fragment>) : part)
        </span>
      )
    }
  }

  {quiz.questions[currentQuestionIndex].type === "multiple-choice" &&
    quiz.questions[currentQuestionIndex].options.map((option, index) => (
      <button
        key={index}
        className={`option-button ${isAnswerSubmitted && option ===
        quiz.questions[currentQuestionIndex].correctAnswer ? 'correct' : isAnswerSubmitted && option !==
        quiz.questions[currentQuestionIndex].correctAnswer ? 'incorrect' : ''}`}
        onClick={() => handleOptionSelect(option)}
        disabled={isAnswerSubmitted} // Disable buttons after answer submission
      >
        {option}
      </button>
    ))
  }

  {quiz.questions[currentQuestionIndex].type === "match-sets" && (
    <MatchSetsQuestion
      question={quiz.questions[currentQuestionIndex]}
      onAnswerSubmit={handleMatchSetsAnswerSubmit}
    />
  )}

  {quiz.questions[currentQuestionIndex].type === "category-sort" && (
    <CategorySortQuestion
      question={quiz.questions[currentQuestionIndex]}
      onAnswerSubmit={handleCategorySortAnswerSubmit}
    />
  )}

```

```
)}

        </div>
    </div>
);
};

export default QuizPage;
```

QuizPage.css

```
.quiz-container {
    text-align: center;
    padding: 20px;
}

.question-text p,
.blank-input,
.option-button,
.next-button{
    margin: auto;
    display: block;
}

.option-button,
.next-button{
    width: 90%;
    max-width: 600px;
    padding: 10px;
    margin-bottom: 10px;
    border: none;
    border-radius: 5px ;
    cursor: pointer ;
    transition-duration: .4s;

    background-color: #007bff ;
    color: white ;
}
```

```

.option-button:hover,
.next-button:hover{
background-color:#0056b3 ;
color:white ;

}

.blank-input{
width :calc(90% - 20px);
max-width :590px;
padding :8px ;
font-size :16px ;

border-radius :5px ;
border :2px solid #007bff ;

}

.correct {
background-color: green !important;
color: white;
}

.incorrect {
background-color: red !important;
color: white;
}

```

Results.js — страница результата

```

import React from 'react';
import { useParams, Link } from 'react-router-dom';
import './Results.css';

const Results = () => {
let { score } = useParams();

return (
<div className="results-container">
<div className="card">
<h2>Ваш счёт</h2>
<p>Вы ответили правильно на {score} Вопросов</p>
<Link to="/">Домой</Link>

```

```
</div>
</div>
);
};


```

```
export default Results;
```

Results.css

```
.results-container {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
}

.card {
    background-color: white;
    padding: 20px;
    border-radius: 10px;
    box-shadow: 0px 4px 8px rgba(0,0,0,0.2);

    width: calc(100% -40px);
    max-width :600px ;
    text-align:center;

}
.card h2,
.card p {
    margin-bottom :20px ;
}

.card a {
    display:inline-block ;
    padding :10px ;
    background-color:#007bff ;
    color:white ;
    text-decoration:none ;
    border-radius :5px ;
    transition-duration:.4s ;

}

.card a:hover{
```

background-color:#0056b3;

color:white ;

}