

Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ
РАБОТА БАКАЛАВРА**

по направлению 09.03.01 «Информатика и
вычислительная техника»
профиль «Разработка программно-
информационных систем»

Разработка автоматизированной обучающей системы для помощи людям с
ограниченными возможностями в изучении жестового языка

Студент

Попов А.Д.

Группа АС-17-1

Руководитель ВКР

Назаркин О.А.

канд. техн. наук, доцент

Консультант по
программному обеспечению

Муравейко А.Ю.

Нормоконтроль

Болдырихин О.В.

Заведующий кафедрой

Алексеев В.А.

канд. техн. наук.

Липецк 2021 г.

Оглавление

Введение	4
1. Постановка задачи	6
1.1. Литературные и патентный обзор постановки подобных задач	6
1.2. Объекты управления, информационные объекты и автоматизируемые процессы. Пользователи и внешние сущности	6
1.3. Цели разработки, функции системы, ограничения и критерии оценки результатов	7
1.3.1. Цели разработки	7
1.3.2. Функции системы	8
1.3.3. Критерии оценки эффективности	8
1.3.4. Ограничения.....	8
2. Изучение и моделирование предметной области	9
2.1. Выявление основных понятий и процессов, их свойств и закономерностей. Построение ER-диаграммы предметной области	9
2.1.1. Основные понятия	9
2.1.2. Диаграмма вариантов использования.....	10
2.1.3. ER-диаграмма предметной области.....	11
2.2. Ключевые сценарии, использования системы	13
2.2.1. Основной сценарий взаимодействия пользователя с системой.....	13
2.2.2. Сценарий взаимодействия пользователя со страницей «Упражнения»	14
2.2.3. Сценарий взаимодействия пользователя со страницей «Тестирование»	14
2.2.4. Сценарий взаимодействия пользователя со страницей «Прогресс»	14
2.2.5. Сценарий добавления новых жестов	15
2.2.6. Сценарий добавления новых категорий	15
2.2.7. Сценарий редактирования данных	15
2.3. Теоретическое изучение предметной области.	15
2.3.1. Упражнения	16
2.3.2. Тестирование	17

2.3.3. Глоссарий	17
3. Разработка информационной базы для решения задачи	18
3.1.1. Архитектура системы	18
3.2. Построение концептуальной и физической модели данных	19
3.2.1. Концептуальная модель данных	19
3.2.2. Физическая модель данных.....	20
3.3. Спецификация	20
3.3.1. Спецификация сущностей.....	20
3.3.1. Спецификация связей	23
3.4. Карта веб-приложения	23
3.4.1. Карта клиентской части.....	23
3.4.2. Карта API	24
4. Программно-аппаратная реализация решения задачи.....	25
4.1. Аппаратное обеспечение.....	25
4.2. Программное обеспечение	25
4.3. Разработанные программные средства	27
4.3.1. Общий алгоритм работы с системой	27
4.3.2. Реализация серверной части	29
4.3.3. Реализация клиентской части	34
4.3.4. Доступ к приложению	40

Введение

Жестовый язык – это самостоятельный язык общения глухих и слабослышащих людей, состоящий из жестов, передаваемых руками в сочетании с выражением лица, движением губ и положением корпуса тела. Первое массовое распространение жестового языка произошло в Европе в учебно-воспитательных центрах для детей во Франции и Германии. Педагогической задачей видели в том, чтобы дети могли овладеть искусственной воссозданным жестовым французским и немецким языками. В России же первая школа обучения жестовому языку возникла в середине XIX века.

В настоящее время в русский жестовый язык на законодательном уровне признан полноценным языком глухих, но до сих пор нельзя сказать, что глухие в достаточной мере могут участвовать в жизни общества. Также в России более 13 миллионов человек с нарушением слуха, из которых более 1 миллиона – это дети. Из всех этих людей всего лишь около 50 тысяч человек родились с этим недугом, остальные же приобрели его в результате какой-либо травмы. Человеку, который столкнулся с данной проблемой нужно адаптироваться к новым обстоятельствам, ему нужно научиться разговаривать «по-новому», то есть выучить жестовый язык. Данное действие такое же не простое, как изучение любого разговорного языка, но тут обучение происходит еще и в атмосфере неблагоприятных обстоятельств.

Жестовые языки являются настолько же непростыми в изучении, как любые звуковые языки. Поэтому на помощь глухим людям приходят сурдопедагоги, обучающие ролики, а также обучающие системы, которые признаны упростить процесс обучения.

Мной была выбрана тема разработки автоматизированной обучающей системы для помощи людям с ограниченными возможностями в изучении жестового языка. Это система будет содержать в себе теоретические материалы, которые в понятной форме смогут разъяснить пользователю значение того или иного жеста. Для закрепления материала пользователю предлагается ряд упражнений, которые будут усложняться и дополняться по мере изучения материала. Также данная

система будет вести статистику успешности прохождения уроков, сколько пользователь прошел уроков, сколько из них были пройдены с первой попытки, какая оценка за них выставлена и т.д. В результате ознакомления с предоставленными ему статистическими данными пользователь сможет обратить внимание на более слабые места в изучении и повторить материал. Для того, чтобы пользователь прошел именно те места, которые изучил хуже всего ему предоставляется возможность создавать собственные коллекции упражнений.

1. Постановка задачи

1.1. Литературные и патентный обзор постановки подобных задач

На данный момент на рынке представлено несколько решений, которые помогают пользователю в обучении, но все они не являются в достаточной мере удобными. Тем более, если нужно обучить языку ребенка, то серое и невзрачное приложение вряд ли сможет его завлечь. Также нет готового решения, которое было бы представлено в виде веб-приложения, то есть программного продукта, который возможно открыть как на компьютере, смартфоне, так и на любом другом устройстве с доступом в интернет.

Ниже представлены некоторые примеры систем для изучения жестового языка для android:

- «Русский жестовый Язык», «Словарь РЖЯ-112». Это два приложения с похожим функционалом. Пользователю показывают ряд изображений с комментариями для изучения букв, слов, после чего ему предлагается закрепить материал, то есть демонстрируют изображение с какой-то определенной буквой или словом и пользователю нужно выбрать правильный ответ.
- «Методы изучения жестового языка». Данное приложение предлагает пользователю набор картинок с изображением жеста и того, что он означает.

1.2. Объекты управления, информационные объекты и автоматизируемые процессы. Пользователи и внешние сущности

Объектом управления является процесс обучения

Автоматизируемые процессы:

- отслеживание ошибок в обучении
- фиксация прогресса изучения материала

Пользователи системы:

- 1) Основной пользователь – человек, который использует систему в целях изучения жестового языка:
 - a. прохождение уроков
 - b. изучение теоретических материалов
 - c. регистрация и авторизация в системе
 - d. просмотр прогресса в изучении материала
 - e. регистрация и авторизация в системе
- 2) Администратор:
 - a. редактирование списка упражнений
 - b. редактирование категорий упражнений
 - c. добавление/удаление медиа-материалов
 - d. авторизация в системе

Человек является внешней сущностью системы. Так как он взаимодействует с системой, то есть он является источником входной информации (администратор) и также является приёмником выходной информации (основной пользователь)

1.3.Цели разработки, функции системы, ограничения и критерии оценки результатов

1.3.1. Цели разработки

Целями создания системы являются:

- упрощение процесса изучения жестового языка
- отслеживание ошибок в обучении
- отслеживание прогресса в изучении материала

1.3.2. Функции системы

- Обеспечение авторизованного доступа в систему
- Добавление новых обучающих материалов в систему
- Удаление и редактирование существующих обучающих материалов
- Прохождение упражнений
- Информирование пользователя о его прогрессе в обучении категорий жестов
- Информирование пользователя об успешности прохождения тестирования
- Авторизованный доступ

1.3.3. Критерии оценки эффективности

- является ли интерфейс удобным и интуитивно понятным.
- смогла ли система сократить время, потраченное на изучение РЖЯ
- корректное функционирование системы

1.3.4. Ограничения

Главным ограничением работы системы является наличие интернет соединения

2. Изучение и моделирование предметной области

2.1. Выявление основных понятий и процессов, их свойств и закономерностей. Построение ER-диаграммы предметной области

2.1.1. Основные понятия

- 1) Жестовый язык — самостоятельный язык, состоящий из жестов, каждый из которых производится руками в сочетании с мимикой, формой или движением рта и губ, а также в сочетании с положением корпуса тела. Эти языки в основном используются в культуре глухих и слабослышащих с целью коммуникации. Использование жестовых языков людьми без нарушения слуха вторично, однако довольно распространено: часто возникает потребность в общении с людьми с нарушениями слуха, являющимися пользователями жестового языка.
- 2) Веб-приложение — клиент-серверное приложение, в котором клиент взаимодействует с веб-сервером при помощи браузера. Логика веб-приложения распределена между сервером и клиентом, хранение данных осуществляется, преимущественно, на сервере, обмен информацией происходит по сети
- 3) Основной пользователь – человек, который использует систему для изучения жестового языка
- 4) Администратор – человек, который добавляет в систему новые данные и редактирует уже имеющиеся
- 5) Упражнение – совокупность определенных жестов и соответствующих им теоретических материалов
- 6) Тестирование – совокупность ранее пройденных пользователем жестов, без демонстрации теоретических материалов.

2.1.2. Диаграмма вариантов использования

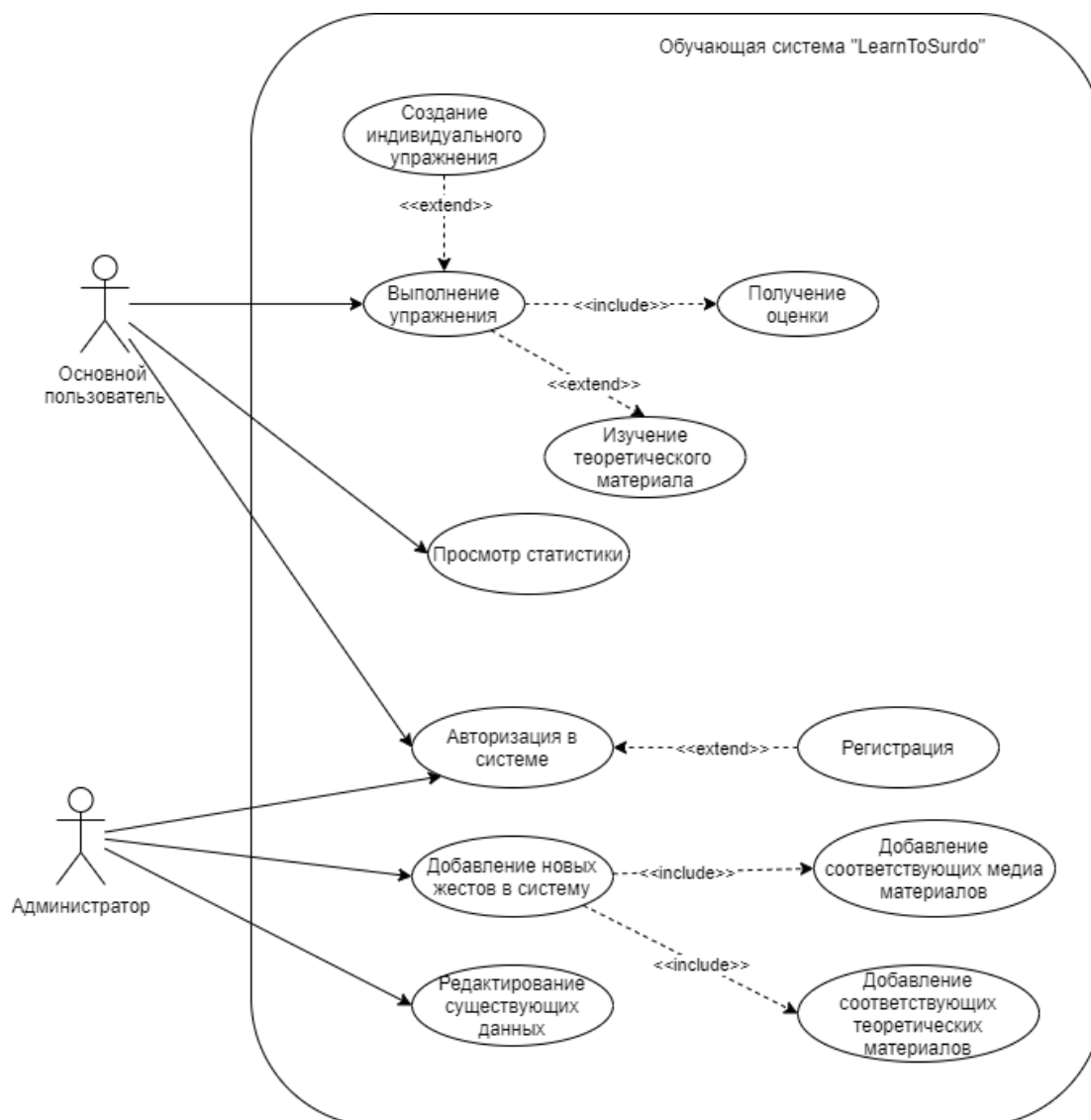


Рисунок 1 - Диаграмма вариантов использования

Основной пользователь:

- Регистрируется и авторизуется в системе
- Выполняет упражнения или тестирование
- Изучает теоретические материалы
- Просматривает прогресс в изучении

Администратор:

- Авторизуется в системе
- Добавляет новые данные в систему

- Редактирует уже существующие.

2.1.3. ER-диаграмма предметной области

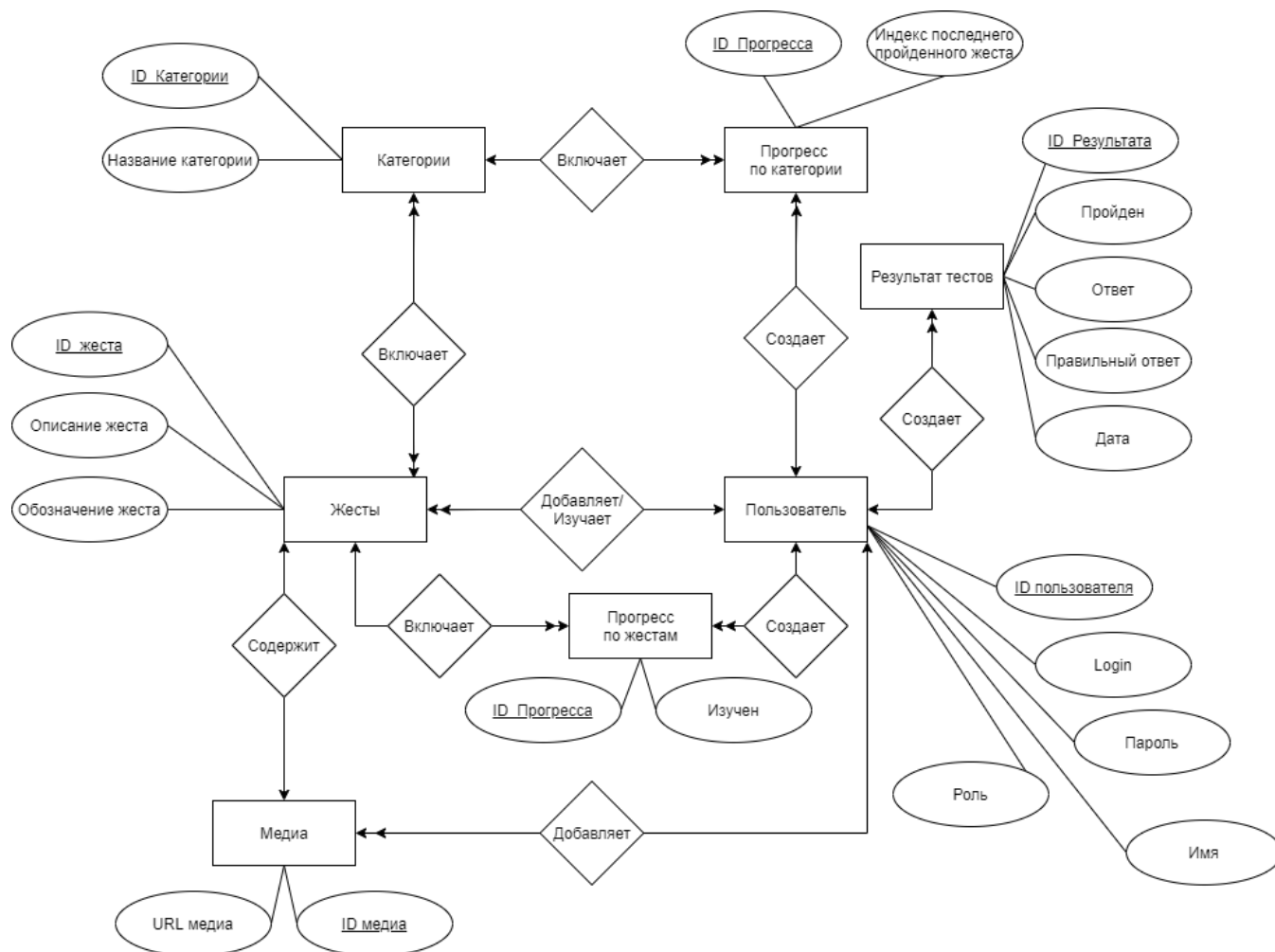


Рисунок 2 - ER-диаграмма в нотации Чена

Таблица 1 - Спецификация сущностей

Имя сущности	Имя атрибута
Пользователь	ID пользователя
	Login
	Пароль
	Имя
	Роль
Жест	ID жеста
	Описание жеста
	Обозначение жеста
Медиа	ID медиа
	URL медиа
Категории	ID категории
	Название категории
Прогресс по категории	ID прогресса
	Индекс последнего пройденного жеста
Прогресс по жестам	ID прогресса
	Изучен
Результат тестов	ID результата
	Пройден
	Ответ
	Правильный ответ
	Дата

- 1) Пользователь: Это или простой человек, который регистрируется в системе. В этой сущности будут храниться данные для авторизации пользователя, его идентификатор, email, пароль в зашифрованном виде и имя, или это администратор: человек, который занимается редактированием данных в системе, удалением и добавлением. Добавляет в систему новые жесты и соответствующие им медиа материалы

- 2) Жест: сущность, в которой хранится описание жеста, который связан с сущностью медиа, категорией, в которой жест находится и с прогрессом по жестам.
- 3) Медиа: сущность, в которой находится URL до картинки или видео жеста.
- 4) Категории: сущность, в которой находится название категории. Эта сущность связана со всеми жестами, прогрессом по категории.
- 5) Прогресс по жестам: сущность, в которой хранится флаг, показывающий, изучен жест или нет.
- 6) Прогресс по категории: сущность, в которой хранится индекс последнего изученного жеста в категории, нужен для того, чтобы сохранять прогресс в изучении категории.
- 7) Результат тестов: сущность, в которой хранится пройденный тест, правильные и неправильные ответы, для того, чтобы позже можно было еще раз просмотреть историю пройденных тестов.

2.2. Ключевые сценарии, использования системы

2.2.1. Основной сценарий взаимодействия пользователя с системой

- 1) Пользователь авторизуется или регистрируется в системе.
- 2) Пользователя переадресовывают на главную страницу.
- 3) Он выбирает страницу, на которую хочет перейти (упражнения, тестирование, прогресс).
- 4) Пользователь взаимодействует со страницей.
- 5) Пользователь выходит из системы.
- 6) Конец.

2.2.2. Сценарий взаимодействия пользователя со страницей

«Упражнения»

- 1) Пользователь переадресовывается на страницу «Упражнения»
- 2) Пользователь выбирает категорию, которую хочет изучить.
- 3) Пользователю демонстрируется 3 теоретических материала, из числа тех, которые не были пройдены ранее.
- 4) После изучения теоретического материала пользователь выполняет 3 вида заданий на закрепление пройденного материала.
- 5) Если пользователь выполнил все верно, то ему демонстрируются следующие 3 теоретических материала и упражнения, если нет, то еще раз проходит те, что выполнил неверно.
- 6) Конец.

2.2.3. Сценарий взаимодействия пользователя со страницей

«Тестирование»

- 1) Пользователь переадресовывается на страницы с тестированием.
- 2) Пользователю предлагается выполнить 10, которые были выбраны случайным образом.
- 3) Пользователь выполняет задания.
- 4) Пользователю показывают количество правильных и неправильных ответов, а также те задания, в которых он допустил ошибку.
- 5) Конец.

2.2.4. Сценарий взаимодействия пользователя со страницей «Прогресс»

- 1) Пользователю переадресовывается на страницу со прогрессом.
- 2) Пользователю показывают прогресс в изучении категорий жестов и результаты пройденных тестов

- 3) Пользователь может нажать на любой из выбранных пройденных тестов, чтобы еще раз просмотреть результат (правильные и неправильные ответы)
- 4) Конец.

2.2.5. Сценарий добавления новых жестов

- 1) Пользователь авторизуется в системе как администратор
- 2) Пользователь вводит в соответствующие поля название жеста и описание. Выбирает категорию, к которой жест относится.
- 3) Нажимает кнопку отправить
- 4) Конец.

2.2.6. Сценарий добавления новых категорий

- 1) Пользователь авторизуется в системе как администратор
- 2) Пользователь вводится в соответствующие поле название категории.
- 3) Нажимает кнопку «Отправить»
- 4) Конец.

2.2.7. Сценарий редактирования данных

- 1) Пользователь авторизуется в системе как администратор.
- 2) Выбирает из таблицы поле, которое нужно изменить.
- 3) Меняет данные и сохраняет.
- 4) Конец.

2.3. Теоретическое изучение предметной области.

В приложении существует 3 варианта изучения жестового языка:

- Упражнения

- Тестирование
- Глоссарий

Внутри первых двух вариантов изучения скрывается еще 3 вида упражнений:

- Упражнение с выбором имени жеста
- Упражнение с выбором изображения жеста
- Упражнение на соответствие

2.3.1. Упражнения

В упражнениях пользователю предлагается выбрать одну из категорий, слова в которой, он хочет изучать. Существуют следующие категории:

- Алфавит
- Числа
- Знакомство
- Человек
- Родство, семья
- Распространенные фразы

Каждая из этих категорий содержит в себе теоретические материалы, в которых описывается жест, а также демонстрируется его изображение, если жест статичный, или видео, если жест происходит в движении.

В упражнениях пользователь сначала изучает теорию, а после закрепляет ее на практике.

После изучения трех теоретических материалов пользователь должен закрепить их. Для каждого жеста пользователь проходит по каждому виду упражнения (выбор названия, выбор изображения и соответствия). Если пользователь в одном из этих видов упражнений допускает 4 ошибки, то придется

пройти его еще раз. В упражнении на соответствие пользователю придется еще раз его проходить, если пользователь в нем допустит хотя бы одну ошибку.

2.3.2. Тестирование

Другой вариант изучения материала – это тестирование. В ней пользователь должен пройти 10 упражнений, вид которых (выбор названия, выбор изображения и соответствие) выбирается случайным образом. После выполнения пользователю показывают результат, где будет выделены правильные ответы и неправильные. Неправильные ответы будут также подкреплены правильным вариантом ответа, чтобы пользователь мог увидеть в чем ошибся.

2.3.3. Глоссарий

Последний вариант изучения материала – это глоссарий. Пользователь может изучить все теоретические материалы по всем категориям без каких-либо ограничений. Для удобства, изученные жесты будут помечены, для того, чтобы пользователь мог отличить уже изученные материалы от неизученных.

3. Разработка информационной базы для решения задачи

3.1.1. Архитектура системы

Перед тем как реализовывать выбранную систему, нужно определиться с выбором платформы, для которой будет она создаваться. Так как очень важно, чтобы обучающая система была кроссплатформенная, выбор пал на реализацию системы в виде веб-приложения с использованием языка JavaScript.

Далее нужно выбрать архитектуру разрабатываемого продукта. Выбор пал на REST архитектуру, которая позволит поддерживать высокую производительность системы и простоту взаимодействия между клиентом и базой данных. Приложение будет работать по принципу клиент-сервер. Взаимодействие с API будет осуществляться по средствам HTTP протокола с помощью методов GET (получение данных), POST (добавление и отправка данных). Для передачи данных будет использоваться формат JSON.

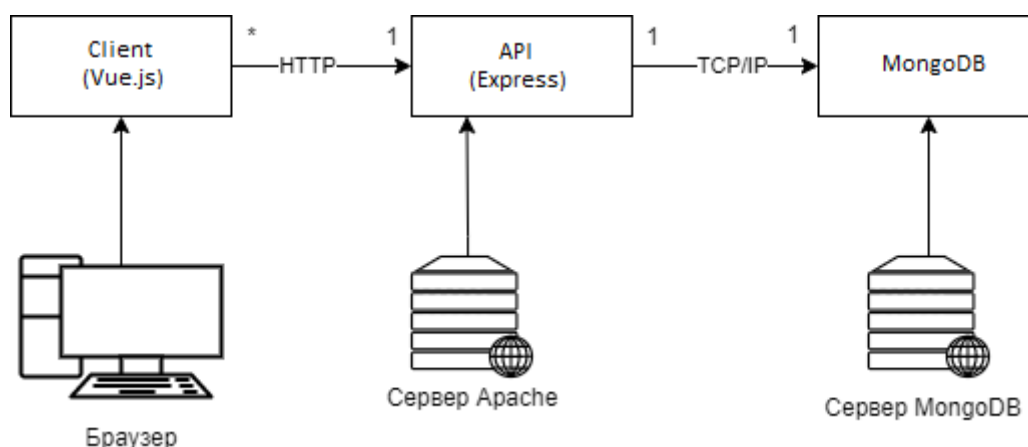


Рисунок 3 - Диаграмма развертывания

3.2. Построение концептуальной и физической модели данных

3.2.1. Концептуальная модель данных

Для веб-приложения необходимо создать базу данных, которая бы хранила в себе всю информацию о жестах, о пользователях и прогрессе пользователей.

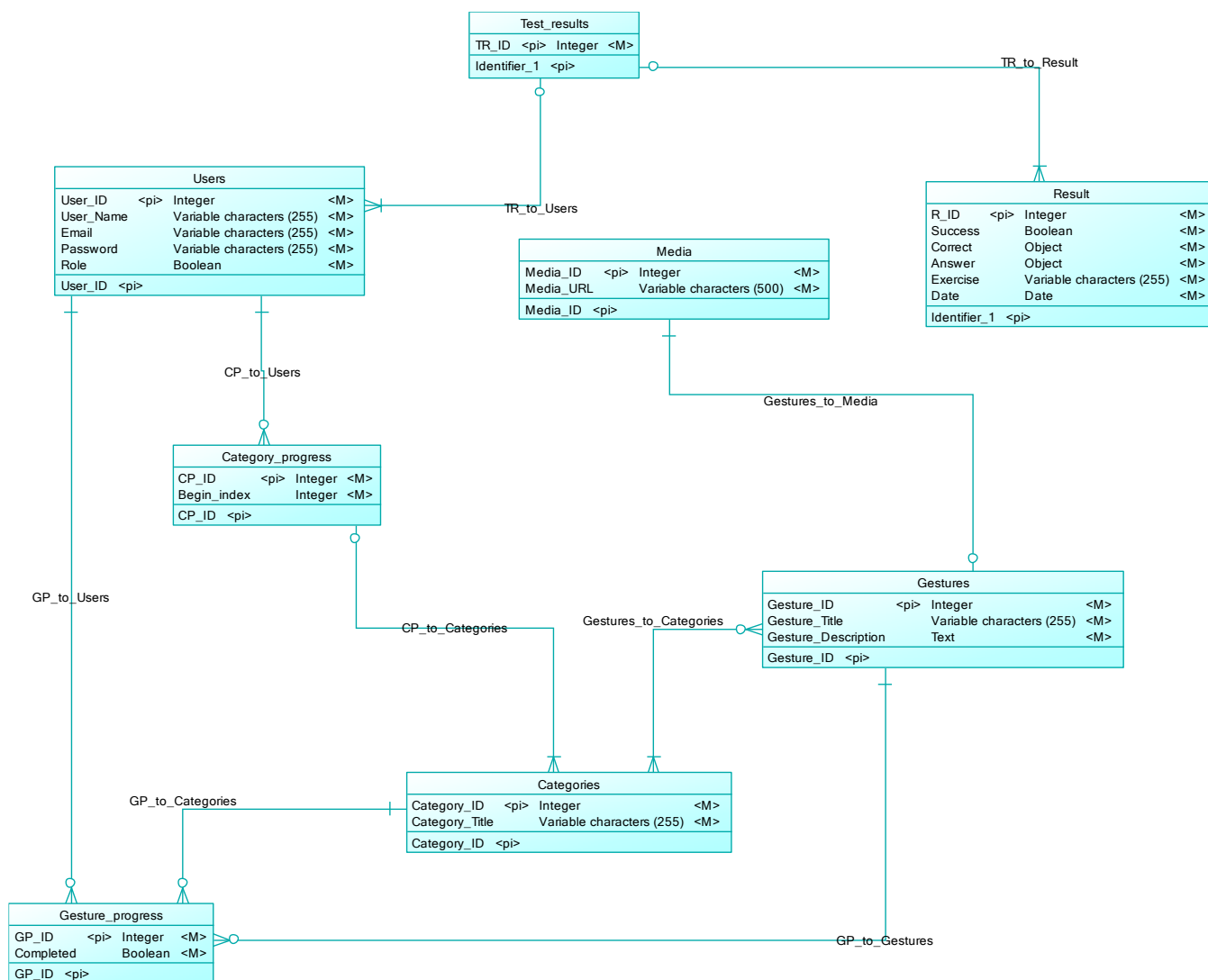


Рисунок 4 - Концептуальная модель данных

3.2.2. Физическая модель данных

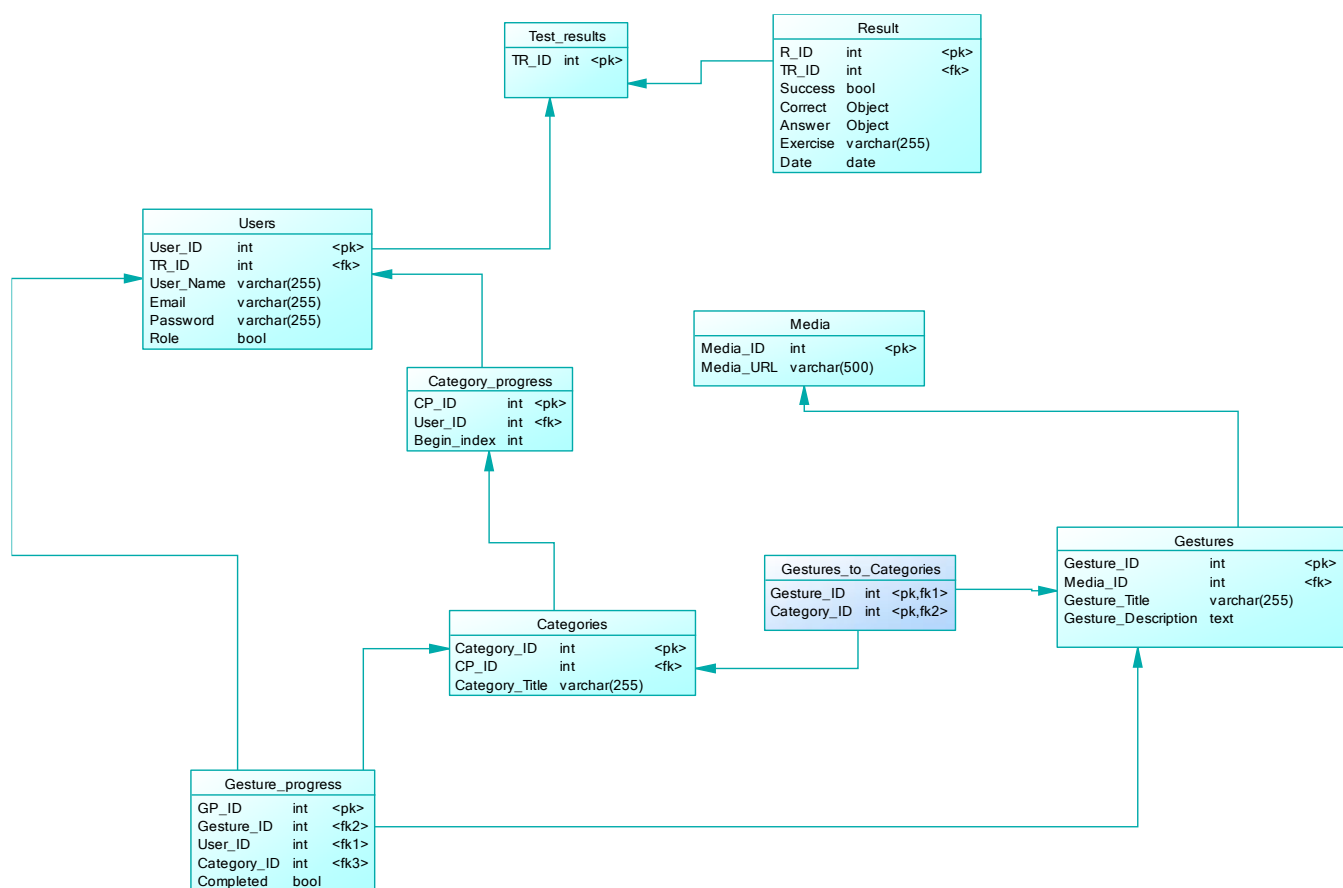


Рисунок 5 - Физическая модель данных

3.3. Спецификация

3.3.1. Спецификация сущностей

Таблица 2 - Спецификация сущности «Пользователи»

Имя атрибута	Расшифровка	Тип данных	Обязательность	Первичный ключ
User_ID	Идентификатор пользователя	Integer	+	+
User_name	Имя пользователя	VarChar(255)	+	-
Login	Логин почта пользователя	VarChar(255)	+	-
Password	Пароль пользователя	VarChar(255)	+	-
Role	Роль пользователя	Boolean	+	-

Таблица 3 - Спецификация сущности «Жесты»

Имя атрибута	Расшифровка	Тип данных	Обязательность	Первичный ключ
Gesture_ID	Идентификатор жеста	Integer	+	+
Gesture_Title	Название жеста	VarChar(255)	+	-
Gesture_Description	Описание жеста	Text	+	-

Таблица 4 - Спецификация сущности «Категории»

Имя атрибута	Расшифровка	Тип данных	Обязательность	Первичный ключ
Category_ID	Идентификатор категории	Integer	+	+
Category_Title	Название категории	VarChar(255)	+	+

Таблица 5 - Спецификация сущности «Медиа»

Имя атрибута	Расшифровка	Тип данных	Обязательность	Первичный ключ
Media_ID	Идентификатор медиа	Integer	+	+
Media_URL	Путь, по которому хранится медиа файл	VarChar(500)	+	-

Таблица 6 - Спецификация сущности «Прогресс категории»

Имя атрибута	Расшифровка	Тип данных	Обязательность	Первичный ключ
CP_ID	Идентификатор «Прогресса категории»	Integer	+	+
Begin_index	Номер последнего изученного жеста в категории	Integer	+	-

Таблица 7 - Спецификация сущности «Прогресс жестов»

Имя атрибута	Расшифровка	Тип данных	Обязательность	Первичный ключ
GP_ID	Идентификатор «Прогресса жестов»	Integer	+	+
Completed	Выполнено	Boolean	+	-

Таблица 8 - Спецификация сущности «Результаты тестов»

Имя атрибута	Расшифровка	Тип данных	Обязательность	Первичный ключ
TR_ID	Идентификатор «Результата тестов»	Integer	+	+

Таблица 9 - Спецификация сущности «Результаты»

Имя атрибута	Расшифровка	Тип данных	Обязательность	Первичный ключ
R_ID	Идентификатор «Результатов»	Integer	+	+
Success	Успешно	Boolean	+	-
Correct	Правильный ответ	Object	+	-
Answer	Ответ	Object	+	-
Exercise	Тип упражнения	VarChar(255)	+	-
Date	Дата выполнения тестирования	Date	+	-

3.3.1. Спецификация связей

Таблица 10 - Спецификация связей

Название	Сущность 1	Сущность 2	Обязательность 1	Обязательность 2	1,2	2,1
TR_to_Users	Test_results	Users	-	+	1:n	0:1
CP_to_Users	Category_progress	Users	-	+	1:1	0:n
GP_to_Users	Gesture_progress	Users	+	-	1:1	0:n
TR_to_Result	Test_results	Result	-	+	1:n	0:1
Gestures_to_Media	Gestures	Media	-	+	1:1	0:1
CP_to_Categories	Category_progress	Categories	-	+	1:n	0:1
Gestures_to_Categories	Gestures	Categories	-	+	1:n	0:n
GP_to_Categories	Gesture_progress	Categories	-	+	1:1	0:n
GP_to_Gestures	Gesture_progress	Gestures	-	+	1:1	0:n

3.4. Карта веб-приложения

3.4.1. Карта клиентской части

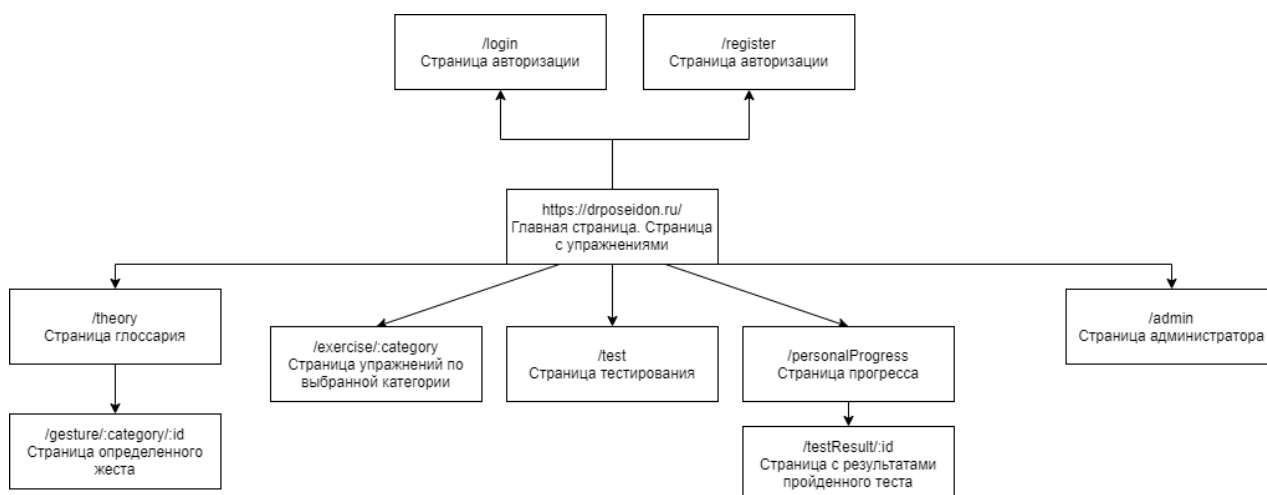


Рисунок 6 - Карта клиентской части веб-приложения

3.4.2. Карта API

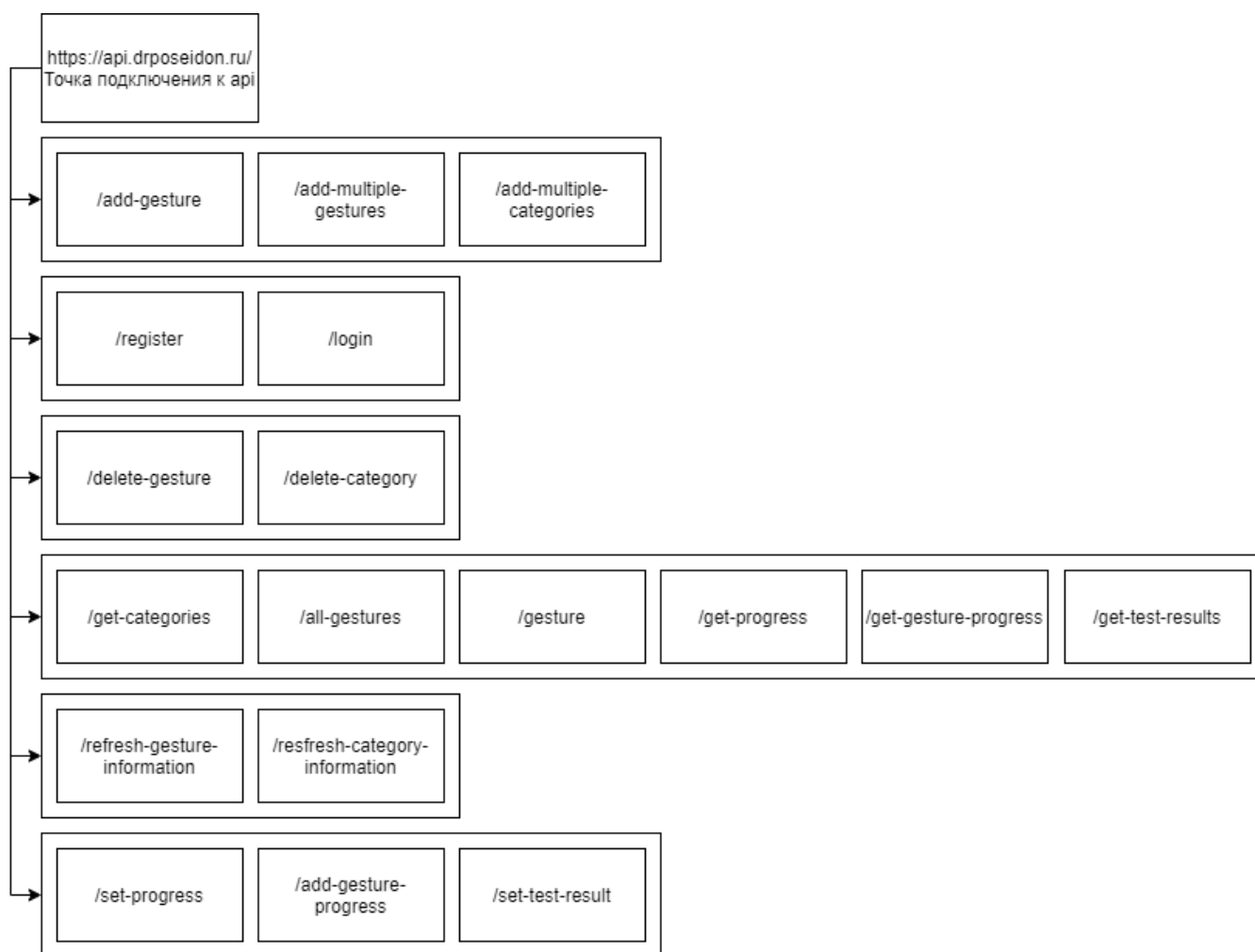


Рисунок 7 - Карта API веб приложения

4. Программно-аппаратная реализация решения задачи

4.1. Аппаратное обеспечение

Для взаимодействия с системой пользователю необходимо иметь любое устройство, поддерживающее веб-браузер и имеющий доступ в Internet

4.2. Программное обеспечение

Для реализации проекта использовались следующие средства:

- Visual Studio Code
- MongoDB
- GIT
- Google Chrome
- Sprinthost

Visual Studio Code – редактор исходного кода, разработанный Microsoft. Позиционируется как «легкий» редактор кода для кроссплатформенной разработки веб и облачных приложений.

MongoDB – документно-ориентированная система управления базами данных, не требующая описания схемы таблиц. Классический пример NoSQL-систем, использует JSON-подобные документы и схему базы данных. Способ хранения данных в MongoDB в этом плане похож на JSON, хотя формально JSON не используется. Для хранения в MongoDB применяется формат, который называется BSON (БиСон) или сокращение от binary JSON.

MongoDB достаточно проста в использовании. Отсутствие жесткой схемы базы данных и в связи с этим потребности при малейшем изменении концепции хранения данных пересоздавать эту схему значительно облегчают работу с базами данных MongoDB и дальнейшим их масштабированием. Кроме того, экономится время разработчиков. Им больше не надо думать о пересоздании базы данных и тратить время на построение сложных запросов.

GIT — это одна из самых известных систем контроля версий с открытым исходным кодом, на которую полагаются миллионы проектов по всему миру (включая как коммерческие, так и бесплатные проекты). Система контроля версий или VCS может значительно облегчить работу разработчиков, пытающихся проанализировать изменения и вклады в общий код. В моем проекте была использована система компании GitHub.

Google Chrome – браузер, разрабатываемый компанией Google. Тестирование и отладка происходила именно в этом браузере, как в версии для ПК, так и для смартфона.

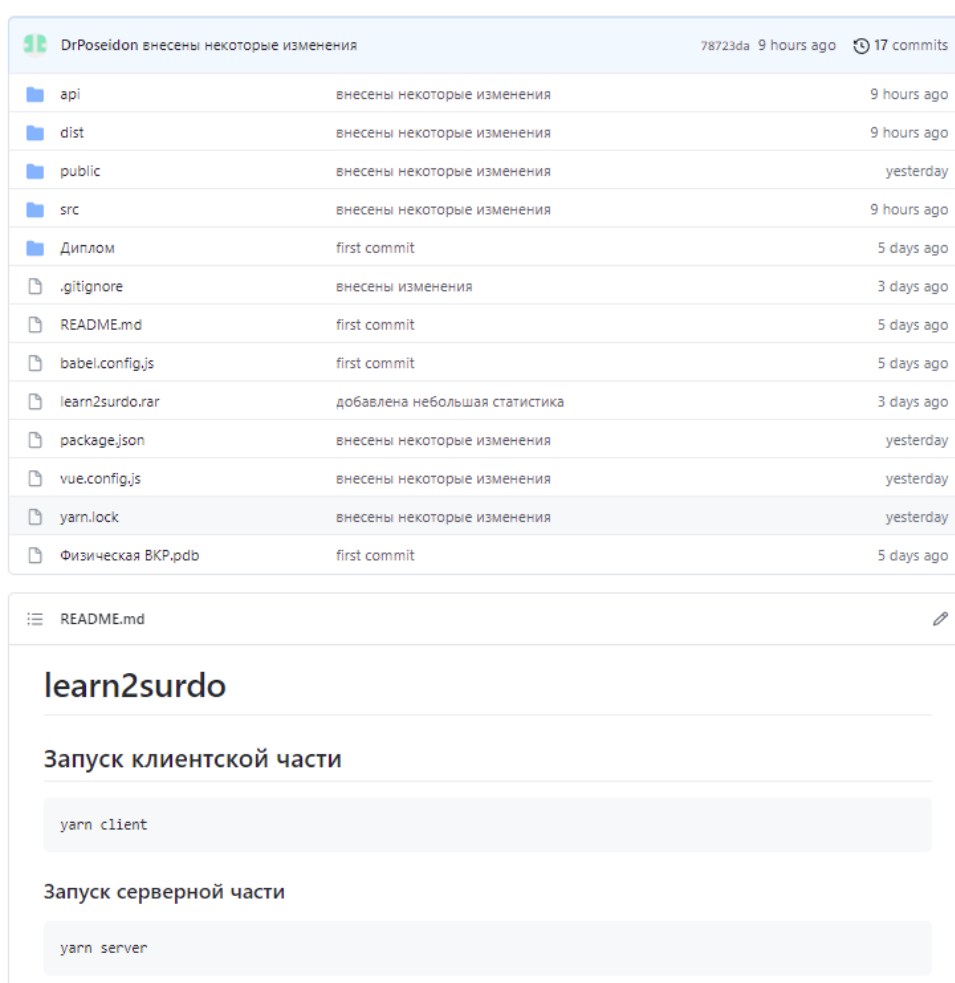


Рисунок 8 - Репозиторий проекта на GitHub

Sprinthost – хостинг, на котором будет размещен проект

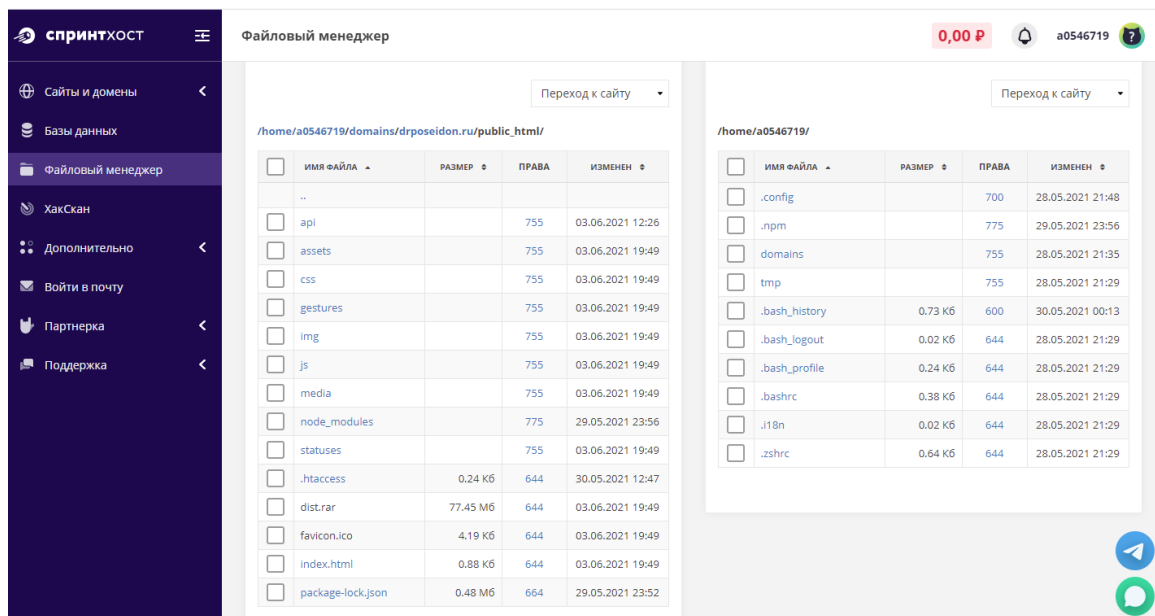


Рисунок 9 - Файловый менеджер проекта на хостинге

4.3. Разработанные программные средства

4.3.1. Общий алгоритм работы с системой

1) Пользователь заходит на сайт и ему предлагается 2 варианта действий: авторизация и регистрация.

- а. Авторизация. Пользователь вводит свой логин и пароль. Если по введенным пользователем данным нет совпадений, то пользователя информируют об этом.
- б. Регистрация. Для регистрации в системе пользователь должен придумать логин, ввести свое имя, пароль, подтверждение пароля. Пароль должен состоять минимум из 6 символов, имя и логин минимум из двух и максимум из двадцати, пароль и подтверждающий пароль должны совпадать, если пользователь не выполнил какие-то из действий или выполнил его неправильно, то система ему об этом сообщит.

После того, как пользователь авторизовался или зарегистрировался, он попадает на главную страницу

- 2) Главная страница. На главной странице представляет собой выбор категории, в которой пользователь желает выполнить упражнения.

Если пользователь уже начал выполнять какую-либо из категорий или полностью ее выполнил, ему об этом сообщит система соответствующим символом и числовым, которое представляет собой дробь, в числителе количество выученных жестов, в знаменателе общее количество жестов.

Если пользователь не выучил ни одного жеста из категории, то будет отображаться только название категории.

По нажатии на выбранную категорию, пользователь попадает на страницу с выполнением упражнений по жестам. Если пользователь уже начал выполнять категорию, или закончил ее выполнять, то пользователю система предложит начать сначала, а при неполном выполнении категории система также предложит продолжить.

- 3) Страница с выполнением упражнений. На странице с упражнениями пользователю демонстрируется 3 теоретических материала по жестам. После их изучения пользователь на каждый из 3х жестов выполняет задание на выбор названия и изображения и одно общее задание на соответствие. Если это был последний блок жестов, то пользователю предлагается пройти категорию еще раз или вернуться на главную страницу. Если это был не последний блок, то пользователю показывают еще 3 жеста и на них он также выполняет задание. Если в категории количество жестов не кратно трем, то к оставшемуся количеству жестов (1 или 2) добавляется последние 1-2 жестов.

Когда пользователь в блоках на выбор названия или изображения допускает 4 ошибки, то ему придется пройти блок заново и так, пока пользователь не сможет пройти его на меньшее количество ошибок.

Правильные ответы пользователю отмечаются зеленым цветом, а неправильные красным

В задании на соответствие, если пользователь допускает 1 ошибку, то придется выполнить это задание заново

- 4) Страница тестирования. На этой странице пользователю демонстрируется 10 случайно случайно выбранных видов упражнений из случайно выбранных жестов. Когда пользователь их выполняет ему демонстрируется результат (правильные и неправильные ответы).
- 5) Страница глоссария. На этой странице пользователю предлагается выбрать категорию, в которой пользователь хочет просмотреть теоретические материалы о жестах. При раскрытии категории пользователю символом галочки обозначают жесты, которые он уже изучил в упражнениях.
- 6) Страница прогресса. На этой странице пользователь может ознакомиться со своим прогрессом в изучении. Он видит графики прогресса по категориям и по тестированию. Если нажать на график с результатами тестирования, то пользователь переходит на просмотр результатов, по выбранному тесту.
- 7) Страница администратора. Данная страница доступна только пользователям, ролью администратора. На этой странице пользователь может добавлять новые категории и жесты, по одному или сразу несколько. Также может просматривать уже добавленные в систему данные и редактировать их.

4.3.2. Реализация серверной части

В качестве базы данных будет использоваться NoSQL (нереляционная) база данных MongoDB. В такой базе данных не используется табличное устройство с четко заданным количеством столбцов и типов данных. В MongoDB центральным понятием является коллекция, который является аналогом таблицы в реляционной базе данных.

Таблица 11 - Сравнение терминологии SQL и NoSQL (MongoDB)

SQL	MongoDB
Таблица	Коллекция
Ряд	Документ
Столбец	Поле
Первичный ключ	ObjectID

Для реализации взаимодействия с базой данных будет использоваться специальная ODM (англ. Object Relational Mapper) библиотека, то есть технология программирования, которая сопоставляет между собой объектную модель данных и базу данных. Это означает, что такой способ позволит определять объекты со строго-типизированной схемой, соответствующей документам базы данных. Для этого была выбрана библиотеку Mongoose. Mongoose – библиотека JavaScript моделирования объектов для MongoDB в среде Node.js.

Mongoose дает возможность создать модель, основанную на определенной схеме. Схема представляет собой объект, поля которого будут являться «столбцами» в коллекции MongoDB.

В проекте использовалось 6 моделей:

В моделях не используется поле первичного ключа, потому что он генерируется в MongoDB при добавлении нового документа в базу данных. Этот ключ называется - `_id`

1) Gestures – модель, описывающая коллекцию gestures. В данной модели использовались поля:

- a. title – название жеста (type String)
- b. description – описание жеста (type String)
- c. category – категория, к которой жест относится (type String)

- d. fileName – название файла, в котором находится изображение или видео жеста (type String)
- 2) Categories – модель, описывающая коллекцию gestures. В данной модели использовалось поле title – название категории (type String).
- 3) GestureProgress – модель, описывающая коллекцию gestureprogresses. В данной модели использовались поля:
- a. userID – идентификатор пользователя (type String)
 - b. gestureID – идентификатор жеста (type String)
 - c. completed – флаг того, что жест выполнен (type Boolean)
 - d. category – название категории, к которой жест относится (type String)
- 4) Progress – модель, описывающая коллекцию progresses. В данной модели использовались поля:
- a. userID – идентификатор пользователя (type String)
 - b. category – название категории (type String)
 - c. beginIndex – номер последнего изученного жеста в категории (type Number)
- 5) TestResults – модель, описывающая коллекцию testresults. В данной модели использовались поля:
- a. userID – идентификатор пользователя (type String)
 - b. results – массив с результатами пройденного теста (type Array)

6) Users – модель, описывающая коллекцию users. В данной модели использовались поля:

- a. login – логин пользователя (type String)
- b. name – имя пользователя (type String)
- c. role – роль пользователя (type Boolean)

На рисунке 10 представлен пример, на котором описывается модель Gestures.

```
const mongoose = require("mongoose");
const Schema = mongoose.Schema;

const gesturesScheme = new Schema(
  {
    title: {
      type: String,
      required: true,
    },
    description: {
      type: String,
      required: true,
    },
    category: {
      type: String,
      required: true,
    },
    fileName: {
      type: String,
      required: true,
    },
  },
  { timestamps: true }
);

const Gesture = mongoose.model("Gestures", gesturesScheme);

module.exports = Gesture;
```

Рисунок 10 – Описание модели Gestures

Для разработки API на Node.js использовался фреймворк Express. Express – это минималистичный и гибкий веб-фреймворк для приложений Node.js, предоставляющий обширный набор функций для разработки веб-приложений. Имя в своем распоряжении множество служебных методов HTTP, то с помощью него можно создать надежный API. На рисунке 11 показана реализация сервера с

помощью express, который будет работать на 80 порту и способен принимать данные в виде JSON.

```
const express = require("express");
const app = express();
const PORT = 80;

app.use(express.urlencoded({ extended: false }));
app.use(express.json());
app.listen(PORT);
```

Рисунок 11 – Развертывание сервера с помощью Express

Для того, чтобы общаться с базой данных, к ней сначала нужно подключиться. Подключение осуществляется с помощью той же библиотеки (Mongoose), что использовалась и для описания моделей данных. На рисунке 12 представлен листинг кода, на котором и происходит подключение.

```
try {
  mongoose.connect(dbConnectionString, {
    useNewUrlParser: true,
    useUnifiedTopology: true,
  });
  console.log("Connection successful");
} catch (err) {
  console.error(err);
}
```

Рисунок 12 – Развертывание сервера с помощью Express

Работа с базой данных будет происходить через GET и POST запросы к API. По средствам нужного метода пользователь будет отправлять данные, с помощью которых сервер, используя модели базы данных, будет отправлять или получать данные из MongoDB. На рисунке 13 представлен листинг кода, на котором по средствам POST запроса пользователь отправляет данные, которые будут использоваться моделью Gestures для обновления информации о каком-либо документе в коллекции gestures.

По адресу <https://api.drposeidon.ru/refresh-gesture-information> пользователь отправляет JSON файл с информацией о жесте. С помощью метода .updateOne объекта Gestures пользователь обновляет данные о жесте с идентификатором _id.

```

app.post("/refresh-gesture-information", async (req, res) => {
  const { _id, title, description, fileName } = req.body;
  if (fileName) {
    await Gestures.updateOne({ _id }, { title, description, fileName })
      .then((result) => {
        res.send(result);
      })
      .catch((err) => {
        console.error(err);
      });
  } else {
    await Gestures.updateOne({ _id }, { title, description })
      .then((result) => {
        res.send(result);
      })
      .catch((err) => {
        console.error(err);
      });
  }
});

```

Рисунок 13 – Развертывание сервера с помощью Express

4.3.3. Реализация клиентской части

Для разработки клиентской части веб-приложения был выбран фреймворк Vue – прогрессивный JavaScript фреймворк для создания пользовательских интерфейсов.

Основными концепциями Vue являются:

- 1) Конструктор
- 2) Компоненты – независимые части системы, которые помогают расширить основные html-элементы.
- 3) Директивы – специальные атрибуты для добавления элементам html дополнительной функциональности.

На рисунке 14 можно увидеть листинг конструктора Vue.js

```

import Vue from "vue";
import App from "./App.vue";
Vue.config.productionTip = false;
new Vue({
  render: (h) => h(App),
}).$mount("#app");

```

Рисунок 14 – Конструктор Vue.js приложения (файл main.js)

4.3.3.1. Маршрутизация в приложении (Vue-router)

Для маршрутизации во Vue.js отвечает библиотека – vue-router. Он помогает в удобной форме настроить роутинг между страницами в веб приложении. Нужно только указать путь и компонент, который за этот путь будет отвечать. На рисунке 15 можно увидеть структуру файла router, в котором прописаны все маршруты в приложении

```
import Vue from "vue";
import VueRouter from "vue-router";
import gesture from "Pages/gesture";
import gestures from "Pages/gestures";
import admin from "Pages/admin";
import exercises from "Pages/exercises";
import exercise from "Pages/exercises/exercise";
import register from "Pages/register";
import login from "Pages/login";
import test from "Pages/test";
import personalProgress from "Pages/personalProgress";
import testResult from "Pages/personalProgress/testResult";

Vue.use(VueRouter);

export default new VueRouter({
  mode: "history",
  routes: [
    { path: "/", component: exercises },
    { path: "/gesture/:category/:id", component: gesture },
    { path: "/theory", component: gestures },
    { path: "/admin", component: admin },
    { path: "/exercise/:category", component: exercise },
    { path: "/register", component: register },
    { path: "/login", component: login },
    { path: "/test", component: test },
    { path: "/personalProgress", component: personalProgress },
    { path: "/testResult/:id", component: testResult },
  ],
});
```

Рисунок 15 – Структура файла router.js, который отвечает за маршрутизацию в приложении

4.3.3.2. Централизованное хранилище (Vuex)

Для управления состоянием в приложении используется централизованное хранилище Vuex. Принцип использования этой библиотеки строится на использовании следующих частей:

- 1) Состояние (State) – источник каких-либо данных приложения.
Изменяет данные в состоянии «Мутация».
- 2) Мутации (Mutations) – то, что может изменять state. Данные для изменения «Состояния» в мутацию попадают из «Действия».
- 3) Действия (Actions) – «мост», между backend API и клиентской частью. «Действие» может как возвращать ответ от бекэнда сразу клиенту, или передавать их дальше в «Мутацию».
- 4) Геттеры (Getters) – методы для получения значений из state.

На рисунке 16 показано то, как взаимодействуют между собой элементы хранилища.

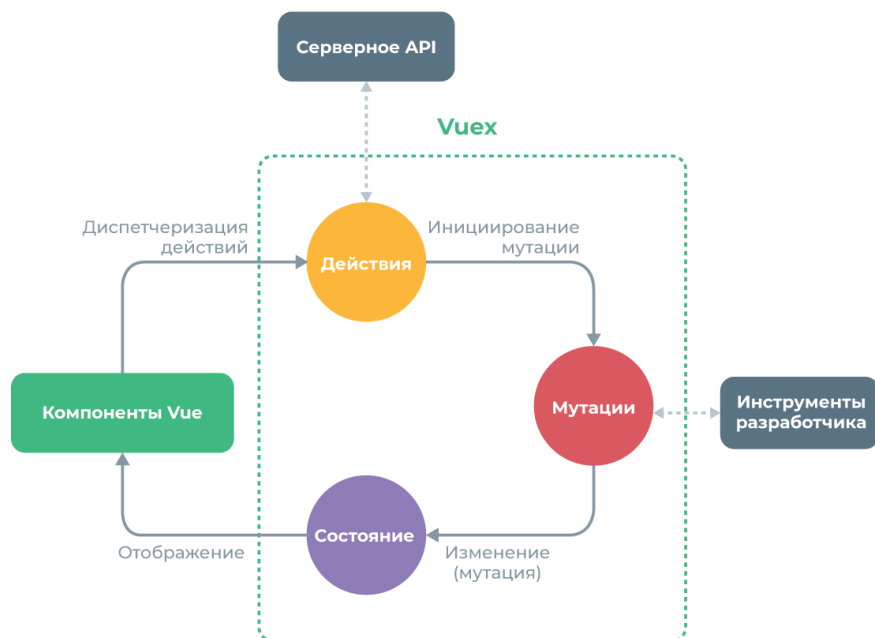


Рисунок 16 - Структура Vuex

В «Действии» происходит все общение между клиентами и базой данных. Для этого общения используется библиотека `axios`, которая позволяет в удобной форме отправлять различные HTTP запросы к бекэнду. На рисунке 17 представлен пример POST запроса на получения результатов тестирования из базы данных, и отправки результата обратно в компонент

```
const GET_TEST_RESULTS = async (vuex, data) => {  
  const res = await axios.post(`${uri}get-test-results`, data);  
  return res.data;  
};
```

Рисунок 17 – Действие GET_TEST_RESULTS

На рисунке 18 представлено «Действие», которое не обращается к базе данных, но взаимодействует с двумя другими действиям (`GET_ALL_CATEGORIES_FROM_DB` и `GET_ALL_GESTURES_FROM_DB`) с помощью функции `dispatch` и потом отправляет результат в «Мутацию» (`SET_GESTURES_BY_CATEGORIES`) для записи данных в «Состояние».

```
const GET_GESTURES_BY_CATEGORIES = async ({ dispatch, commit, state }) => {  
  await dispatch("GET_ALL_CATEGORIES_FROM_DB");  
  await dispatch("GET_ALL_GESTURES_FROM_DB");  
  let newGestures = [];  
  const { categories, gestures } = state;  
  for (let i = 0; i < categories.length; i++) {  
    let tmp = [];  
    for (let j = 0; j < gestures.length; j++) {  
      if (gestures[j].category === categories[i].title) {  
        tmp.push(gestures[j]);  
      }  
    }  
    newGestures.push(tmp);  
  }  
  commit("SET_GESTURES_BY_CATEGORIES", newGestures);  
};
```

Рисунок 18 – Действие GET_GESTURES_BY_CATEGORIES

4.3.3.3. Элементы веб-приложения

Компоненты, которые представляют собой страницы, которые использовались как входные точки в маршрутизации:

- 1) register – компонент, являющийся основным для страницы регистрации, реализующий интерфейс для регистрации пользователя. Логика валидации форм и сбор информации для отправки в базу данных
- 2) login – компонент, являющийся основным для страницы авторизации, реализующий интерфейс авторизации пользователя.
- 3) admin – компонент, являющийся основным для страницы администратора. Включающий другие компоненты:
 - a. L2SAddForm – отвечает за логику добавления новых данных
 - b. L2SShowAllData – отвечает за логику получения данных из базы данных и отображения их на странице в виде таблиц, с возможностью редактирования в них информации.
- 4) exercises – компонент, являющийся основным для главной страницы, где можно выбрать категорию, в которой пользователь желает выполнить упражнения
- 5) exercise – компонент, являющийся страницей, на которой расположены упражнения по выбранной категории. Страница, на которую можно перейти, нажав на нужную категорию на странице exercises. Этот компонент включает в себя другие:
 - a. L2SComparisonTask – отвечает за интерфейс и логику упражнения на соответствие
 - b. L2SImageSelectionTask – отвечает за интерфейс и логику упражнения на выбор изображения, по названию жеста
 - c. L2SNameSelectionTask – отвечает за интерфейс и логику упражнения на выбор имени жеста, по его изображению

- d. L2Sexercise – отвечает за отображение компонентов L2SComparisonTask, L2SImageSelectionTask, L2SNameSelectionTask и логику, связанную с их отображением
 - e. L2STheory – отвечает за отображение теоретических материалов
 - f. L2SProgressBar – отвечает за отображение полосы прогресса по категории
- 6) test – компонент, являющийся страницей, который реализует логику формирования теста. Этот компонент включает в себя и другие
- a. L2STestComparisonTask – тестовый вариант задания на соответствие
 - b. L2STestImageSelectionTask – тестовый вариант задания на выбор изображения
 - c. L2SNameSelectionTask – тестовый вариант задания на выбор названия
 - d. Exercises – компонент, который содержит в себе 3 компонента, отвечающих за задания, описанные выше
 - e. Comparison, ImageSelection, NameSelection – компоненты, отвечающие за отображение результатов по упражнениям на соответствие, выбор изображения и выбор имени
- 7) gestures – компонент, являющийся страницей, расположены с кнопки, являющиеся выпадающими списками с выбором теоретических материалов по категории
- 8) gesture – компонент, отвечающий за отображение теоретических материалов по категории
- 9) personalProgress – компонент, являющийся странице с отображением прогресса пользователя по упражнениям и по тестам

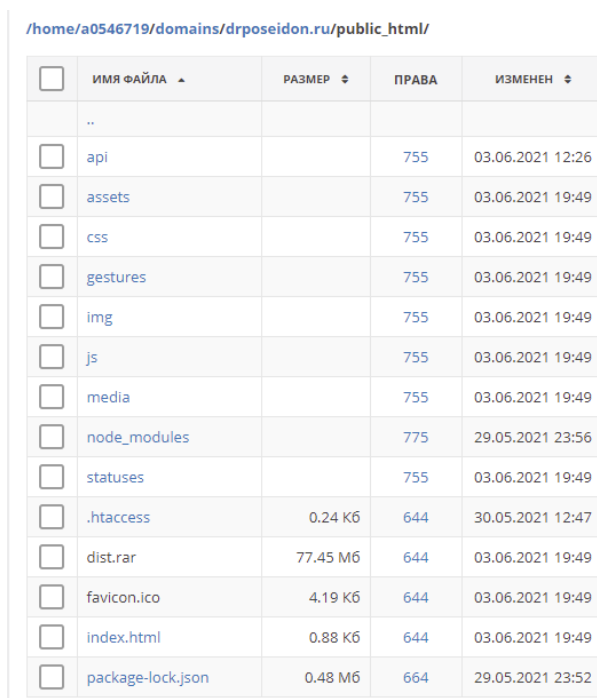
Ниже представлены компоненты, которые вынесены отдельно, так как используются на нескольких страницах

- 1) L2SHeader – компонент, отвечающий за отображение хедера на страницах
- 2) L2SMedia – компонент, отвечающий за отображение изображения или видео с жестом
- 3) L2Stable – компонент, отвечающий за построение таблиц

4.3.4. Доступ к приложению

Для того, чтобы пользователь мог без труда пользоваться веб-приложением с любого устройства с доступом в интернет – его нужно развернуть на хостинге, выбранный хостинг работает с HTTP сервером Apache.

Для этого нужно production версию приложения разместить в корневой папке домена (рисунок 19)



<input type="checkbox"/>	ИМЯ ФАЙЛА	РАЗМЕР	ПРАВА	ИЗМЕНЕН
	..			
<input type="checkbox"/>	api		755	03.06.2021 12:26
<input type="checkbox"/>	assets		755	03.06.2021 19:49
<input type="checkbox"/>	css		755	03.06.2021 19:49
<input type="checkbox"/>	gestures		755	03.06.2021 19:49
<input type="checkbox"/>	img		755	03.06.2021 19:49
<input type="checkbox"/>	js		755	03.06.2021 19:49
<input type="checkbox"/>	media		755	03.06.2021 19:49
<input type="checkbox"/>	node_modules		775	29.05.2021 23:56
<input type="checkbox"/>	statuses		755	03.06.2021 19:49
<input type="checkbox"/>	.htaccess	0.24 Кб	644	30.05.2021 12:47
<input type="checkbox"/>	dist.rar	77.45 Мб	644	03.06.2021 19:49
<input type="checkbox"/>	favicon.ico	4.19 Кб	644	03.06.2021 19:49
<input type="checkbox"/>	index.html	0.88 Кб	644	03.06.2021 19:49
<input type="checkbox"/>	package-lock.json	0.48 Мб	664	29.05.2021 23:52

Рисунок 19 - Файлы клиентской части веб-приложения, расположенные на хостинге

Чтобы приложение корректно работало, нужно создать файл .htaccess, в который нужно добавить информацию о том, чтобы по адрес всегда был доступен по защищенному HTTP протоколу (HTTPS) и чтобы точкой входа для приложения был файл index.html

```
RewriteEngine on
RewriteCond %{HTTP:X-Forwarded-Proto} !https
RewriteRule ^ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301,NE]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule . index.html
```

Рисунок 20 - Файл .htaccess для клиентской части

Файлы бекэнда нужно расположить в поддомене, для этого была создана в папке с проектом папка api, в которую были помещены все файлы, отвечающие за бекэнд.

/home/a0546719/domains/drposeidon.ru/public_html/api/

<input type="checkbox"/>	ИМЯ ФАЙЛА ▲	РАЗМЕР ⇅	ПРАВА	ИЗМЕНЕН ⇅
	..			
<input type="checkbox"/>	models		755	03.06.2021 12:26
<input type="checkbox"/>	routes		755	03.06.2021 12:26
<input type="checkbox"/>	.htaccess	0.37 Kб	644	29.05.2021 23:57
<input type="checkbox"/>	api.rar	3.41 Kб	644	03.06.2021 12:25
<input type="checkbox"/>	app.js	0.28 Kб	644	03.06.2021 12:26

Рисунок 21 - Файлы backend части веб-приложения, расположенные на хостинге

Также, как и для клиентской части, нужно описать .htaccess файл. Также нужно добаавить информацию о том, чтобы бекэнд всегда был доступен по защищенному протоколу (HTTPS) и чтобы точкой входа был файл app.js

```
RewriteEngine on
RewriteCond %{HTTP:X-Forwarded-Proto} !https
RewriteRule ^ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301,NE]

SetEnv GHOST_NODE_VERSION_CHECK false
PassengerStartupFile app.js
PassengerResolveSymlinksInDocumentRoot on
Require all granted
PassengerAppType node
PassengerAppRoot /home/a0546719/domains/drposeidon.ru/public_html/api/
Options -MultiViews
```

Рисунок 22 - Файл .htaccess для backend части

В результате этих действий, по адресу drposeidon.ru стало доступно любому пользователю веб-приложение Learn2Surdo