

Отчет о проверке на заимствования №1



Автор: Ларин Михаил Олегович

Проверяющий: Ларин Михаил (hanter715@gmail.com / ID: 8663704)

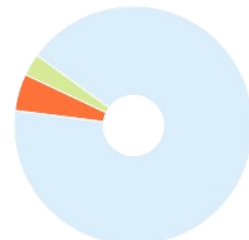
Отчет предоставлен сервисом «Антиплагиат» - <http://users.antiplagiat.ru>

ИНФОРМАЦИЯ О ДОКУМЕНТЕ

№ документа: 2
Начало загрузки: 10.01.2021 12:01:42
Длительность загрузки: 00:00:15
Имя исходного файла: B17-V71_LarinMO_VKR_PZ_v7_1.pdf
Название документа: B17-V71_LarinMO_VKR_PZ_v7_1
Размер текста: 1 кБ
Символов в тексте: 75589
Слов в тексте: 8625
Число предложений: 607

ИНФОРМАЦИЯ ОБ ОТЧЕТЕ

Последний готовый отчет (ред.)
Начало проверки: 10.01.2021 12:01:58
Длительность проверки: 00:00:24
Комментарии: не указано
Поиск перефразирований: да
Модули поиска: Модуль поиска ИПС "Адилет", Модуль выделения библиографических записей, Сводная коллекция ЭБС, Модуль поиска Интернет плюс, Коллекция РГБ, Цитирование, Переводные заимствования (RuEn), Модуль поиска переводных заимствований по elibrary (EnRu), Модуль поиска переводных заимствований по elibrary (KkRu), Модуль поиска переводных заимствований по elibrary (KyRu), Модуль поиска переводных заимствований по интернет (EnRu), Модуль поиска переводных заимствований по интернет (KkRu), Модуль поиска переводных заимствований по интернет (KyRu), Модуль поиска переводных заимствований (KkEn), Коллекция eLIBRARY.RU, Коллекция Гарант, Модуль поиска Интернет, Коллекция Медицина, Модуль поиска перефразирований eLIBRARY.RU, Модуль поиска перефразирований Интернет, Коллекция Патенты, Модуль поиска общеупотребительных выражений, Кольцо вузов, Переводные заимствования



ЗАИМСТВОВАНИЯ

5,39%

САМОЦИТИРОВАНИЯ

0%

ЦИТИРОВАНИЯ

2,64%

ОРИГИНАЛЬНОСТЬ

91,97%

Заимствования — доля всех найденных текстовых пересечений, за исключением тех, которые система отнесла к цитированиям, по отношению к общему объему документа.
Самоцитирования — доля фрагментов текста проверяемого документа, совпадающий или почти совпадающий с фрагментом текста источника, автором или соавтором которого является автор проверяемого документа, по отношению к общему объему документа.
Цитирования — доля текстовых пересечений, которые не являются авторскими, но система посчитала их использование корректным, по отношению к общему объему документа. Сюда относятся оформленные по ГОСТу цитаты; общеупотребительные выражения; фрагменты текста, найденные в источниках из коллекций нормативно-правовой документации.
Текстовое пересечение — фрагмент текста проверяемого документа, совпадающий или почти совпадающий с фрагментом текста источника.
Источник — документ, проиндексированный в системе и содержащийся в модуле поиска, по которому проводится проверка.
Оригинальность — доля фрагментов текста проверяемого документа, не обнаруженных ни в одном источнике, по которым шла проверка, по отношению к общему объему документа.
Заимствования, самоцитирования, цитирования и оригинальность являются отдельными показателями и в сумме дают 100%, что соответствует всему тексту проверяемого документа. Обращаем Ваше внимание, что система находит текстовые пересечения проверяемого документа с проиндексированными в системе текстовыми источниками. При этом система является вспомогательным инструментом, определение корректности и правомерности заимствований или цитирований, а также авторства текстовых фрагментов проверяемого документа остается в компетенции проверяющего.

№	Доля в отчете	Источник	Ссылка	Актуален на	Модуль поиска
[01]	2%	не указано	не указано	раньше 2011	Модуль выделения библиографических записей
[02]	0,85%	2017_ИЗИТУС_ПОВТАС_090304_БР_Темников_Никита_Александрович.docx	не указано	21 Июн 2017	Кольцо вузов
[03]	0,15%	Дементьев Антон Михайлович_ИП-15-М	не указано	31 Мая 2017	Кольцо вузов
[04]	0%	Java - Howling Pixel	https://howlingpixel.com	13 Мая 2019	Модуль поиска Интернет
[05]	0%	Дипломная работа Чекмарева Алексея (доработка).docx	не указано	15 Июн 2020	Кольцо вузов
[06]	1,19%	Java Virtual Machine	http://ru.wikipedia.org	05 Янв 2017	Модуль поиска перефразирований Интернет
[07]	0%	Java Virtual Machine	http://ru.wikipedia.org	14 Фев 2018	Модуль поиска Интернет
[08]	0%	Java Virtual Machine	http://ru.wikipedia.org	20 Авг 2020	Модуль поиска Интернет
[09]	0%	Java Virtual Machine — Википедия	https://ru.wikipedia.org	05 Мая 2020	Модуль поиска Интернет
[10]	0%	ВКРБерезина	не указано	21 Июн 2017	Кольцо вузов
[11]	0%	Полная версия научной работы 228 КБ	http://scienceforum.ru	04 Мая 2018	Модуль поиска Интернет
[12]	0,79%	АВТОМАТИЗИРОВАННАЯ СИСТЕМА КЛАССИФИКАЦИИ МУЗЫКАЛЬНЫХ ЗАПИСЕЙ ...	http://elibrary.ru	15 Янв 2019	Модуль поиска перефразирований eLIBRARY.RU
[13]	0%	2 МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ Федераль...	http://netess.ru	04 Дек 2016	Модуль поиска Интернет
[14]	0%	ФИСТ_ПОВТиАС_ИСТ_Перезнатнов_Антон_Васильевич.doc	не указано	11 Фев 2014	Кольцо вузов

[15]	0%	Java Virtual Machine	http://ru.wikipedia.org	13 Ноя 2016	Модуль поиска Интернет
[16]	0,65%	не указано	не указано	раньше 2011	Модуль поиска общеупотребительных выражений
[17]	0,41%	ОБЗОР КЛИЕНТСКОЙ ЧАСТИ МОБИЛЬНОГО ПРИЛОЖЕНИЯ TRANSPORTTV.	http://elibrary.ru	14 Янв 2020	Модуль поиска перефразирований eLIBRARY.RU
[18]	0,06%	ОБЗОР КЛИЕНТСКОЙ ЧАСТИ МОБИЛЬНОГО ПРИЛОЖЕНИЯ TRANSPORTTV.	http://elibrary.ru	14 Янв 2020	Коллекция eLIBRARY.RU
[19]	0,06%	Java Virtual Machine — Википедия	https://ru.wikipedia.org	10 Янв 2021	Модуль поиска Интернет плюс
[20]	0%	Java Virtual Machine — Википедия	https://ru.wikipedia.org	10 Янв 2021	Модуль поиска Интернет плюс
[21]	0%	Java Virtual Machine — Википедия	https://ru.wikipedia.org	10 Янв 2021	Модуль поиска Интернет плюс
[22]	0,33%	Программа учета заводских номеров изготовленных изделий на основе технолог...	http://library.eltech.ru	19 Сен 2019	Модуль поиска Интернет
[23]	0%	Разработка мобильного приложения для демонстрации применения методов иск...	http://elib2.altstu.ru	29 Янв 2017	Модуль поиска перефразирований Интернет
[24]	0%	ФИСТ_ПОВТиАС_ИСТ_ДР_Чекулаев_Сергей_Андреевич.doc	не указано	11 Фев 2014	Кольцо вузов
[25]	0%	2016BKP230607ТИТКОВ.docx	не указано	01 Июн 2016	Кольцо вузов
[26]	0%	ДИПЛОМНАЯ РАБОТА Ключинская-Кизицкая 1	не указано	12 Янв 2018	Кольцо вузов
[27]	0%	Программа учета заводских номеров изготовленных изделий на основе технолог...	http://library.eltech.ru	19 Сен 2019	Модуль поиска Интернет плюс
[28]	0%	Архитектурные решения java для доступа к данным.	http://elibrary.ru	11 Мар 2020	Коллекция eLIBRARY.RU
[29]	0%	часть 4 (1/2)	https://belstu.by	07 Ноя 2018	Модуль поиска Интернет
[30]	0,13%	2017_M_ИВТ_ИВТ_МД_Андреев_Александр_Геннадьевич.>	https://psuti.ru	10 Авг 2018	Модуль поиска Интернет
[31]	0%	Макет ЧАСТЬ 2 2017.pdf	http://docs.gsu.by	10 Ноя 2017	Модуль поиска Интернет
[32]	0%	Java Virtual Machine	http://ru.wikipedia.org	14 Фев 2018	Модуль поиска Интернет плюс
[33]	0%	Java - Howling Pixel	https://howlingpixel.com	13 Мая 2019	Модуль поиска Интернет плюс
[34]	0,17%	Шунько Анна Андреевна дипломная_работа_ПИ_4_Шунько.docx	не указано	14 Мая 2017	Кольцо вузов
[35]	0,37%	Spring MVC — основные принципы / Хабр	https://habr.com	08 Апр 2020	Модуль поиска Интернет
[36]	0,01%	Анализ фреймворков для создания web-приложений на Java.	http://elibrary.ru	24 Янв 2020	Модуль поиска перефразирований eLIBRARY.RU
[37]	0%	Архитектурные решения java для доступа к данным.	http://elibrary.ru	11 Мар 2020	Модуль поиска перефразирований eLIBRARY.RU
[38]	0%	Spring MVC — основные принципы / Хабр	https://habr.com	08 Апр 2020	Модуль поиска Интернет плюс
[39]	0%	Spring MVC — основные принципы / Хабр	https://habr.com	10 Янв 2021	Модуль поиска Интернет плюс
[40]	0%	Анализ фреймворков для создания web-приложений на Java.	http://elibrary.ru	24 Янв 2020	Коллекция eLIBRARY.RU
[41]	0%	Apache Tomcat — Википедия	https://ru.wikipedia.org	04 Апр 2019	Модуль поиска Интернет
[42]	0%	Apache Tomcat - это... Что такое Apache Tomcat?	https://dic.academic.ru	04 Апр 2019	Модуль поиска Интернет
[43]	0%	Apache Tomcat — Википедия	https://ru.wikipedia.org	20 Июн 2019	Модуль поиска Интернет
[44]	0%	Мазурова, Ирина Сергеевна Численные методы построения оптимального управ...	http://dlib.rsl.ru	22 Авг 2019	Коллекция РГБ
[45]	0%	ТЕХНОЛОГИИ ВИРТУАЛИЗАЦИИ И ВИРТУАЛЬНЫЕ МАШИНЫ.	http://elibrary.ru	31 Дек 2016	Коллекция eLIBRARY.RU
[46]	0%	Apache Tomcat — Википедия	https://ru.wikipedia.org	20 Июн 2019	Модуль поиска Интернет плюс
[47]	0%	Apache Tomcat — Википедия	https://ru.wikipedia.org	10 Янв 2021	Модуль поиска Интернет плюс
[48]	0%	Apache Tomcat — Википедия	https://ru.wikipedia.org	10 Янв 2021	Модуль поиска Интернет плюс
[49]	0%	Организация доступа к реляционной базе данных на основе технологии ORM с ис...	https://moluch.ru	10 Янв 2021	Модуль поиска Интернет плюс
[50]	0%	Тармин, Виктор Анатольевич диссертация ... кандидата технических наук : 05.13.06...	http://dlib.rsl.ru	01 Янв 2010	Коллекция РГБ
[51]	0%	Овечкин, Роман Михайлович диссертация ... кандидата технических наук : 05.13.10...	http://dlib.rsl.ru	15 Сен 2015	Коллекция РГБ
[52]	0%	Разработка автоматизированной информационной системы "Аукционы и тендер...	http://elibrary.ru	26 Окт 2019	Модуль поиска перефразирований eLIBRARY.RU
[53]	0%	https://rk6.bmstu.ru/pub/diplom_labors/2018/2018_Shaturnov_A_rpz.pdf	https://rk6.bmstu.ru	14 Мая 2019	Модуль поиска Интернет плюс
[54]	0%	WEBSOCKET КАК ИНСТРУМЕНТ ДЛЯ УДАЛЁННОГО УПРАВЛЕНИЯ.	http://elibrary.ru	14 Янв 2019	Коллекция eLIBRARY.RU

[55]	0%	https://rk6.bmstu.ru/pub/diplom_labors/2018/2018_Shaturnov_A_rpz.pdf	https://rk6.bmstu.ru	14 Мая 2019	Модуль поиска Интернет
[56]	0%	часть 4 (1/2)	https://belstu.by	07 Ноя 2018	Модуль поиска Интернет плюс
[57]	0%	The OpenNET Project: Ссылки на программное обеспечение (базовая разбивка)	http://opennet.ru	13 Мая 2019	Модуль поиска Интернет
[58]	0%	The OpenNET Project: Ссылки на программное обеспечение (базовая разбивка)	http://opennet.ru	13 Мая 2019	Модуль поиска Интернет плюс
[59]	0%	Java — Википедия	https://ru.wikipedia.org	26 Мая 2020	Модуль поиска Интернет плюс
[60]	0%	Java — Википедия	https://ru.wikipedia.org	10 Янв 2021	Модуль поиска Интернет плюс
[61]	0,23%	Диплом - Логинов И.Д. Без Титульника и Списка источников.docx	не указано	07 Июн 2020	Кольцо вузов
[62]	0%	WEB-ориентированная среда решения оптимизационных задач в транспортной с...	http://elibrary.ru	14 Сен 2015	Коллекция eLIBRARY.RU
[63]	0%	Автоматическое распознавание голосовых сообщений в мессенджерах с использ...	http://elibrary.ru	21 Апр 2020	Коллекция eLIBRARY.RU
[64]	0%	Изучаем Retrofit 2 / Хабр	https://habr.com	10 Янв 2021	Модуль поиска Интернет плюс
[65]	0,21%	Диссертация	http://spiiaras.nw.ru	29 Янв 2017	Модуль поиска перефразирований Интернет
[66]	0%	Макет ЧАСТЬ 2 2017.pdf	http://docs.gsu.by	10 Ноя 2017	Модуль поиска Интернет плюс
[67]	0%	Скачать	http://worldreferat.ru	27 Окт 2018	Модуль поиска Интернет плюс
[68]	0%	Скачать	http://worldreferat.ru	27 Окт 2018	Модуль поиска Интернет
[69]	0,18%	МОДЕЛИРОВАНИЕ И СОЗДАНИЕ ВИРТУАЛЬНЫХ МОДЕЛЕЙ ОБЪЕКТОВ ЖЕЛЕЗНОД...	http://elibrary.ru	26 Янв 2020	Модуль поиска перефразирований eLIBRARY.RU
[70]	0%	АВТОМАТИЗИРОВАННАЯ ИНФОРМАЦИОННАЯ СИСТЕМА УЧЕТА И МОНИТОРИНГ...	http://elibrary.ru	15 Янв 2018	Коллекция eLIBRARY.RU
[71]	0%	Пестун, Максим Вадимович Методы построения навигационных описаний марш...	http://dlib.rsl.ru	27 Дек 2019	Коллекция РГБ
[72]	0%	http://socioprognoz-ru.1gb.ru/files/File/2018/Arefiev_Sbornik_8_001_536_2018_ispr8_1...	http://socioprognoz-ru.1gb.ru	04 Дек 2020	Модуль поиска Интернет плюс
[73]	0%	http://socioprognoz-ru.1gb.ru/files/File/2018/Arefiev_Sbornik_8_001_536_2018_ispr8_1...	http://socioprognoz-ru.1gb.ru	04 Дек 2020	Модуль поиска Интернет плюс
[74]	0%	Организовываем взаимодействие между ПК и ЦАП/АЦП при помощи ПЛИС / Хабр	https://habr.com	10 Янв 2021	Модуль поиска Интернет плюс
[75]	0%	Организовываем взаимодействие между ПК и ЦАП/АЦП при помощи ПЛИС / Хабр	https://habr.com	10 Янв 2021	Модуль поиска Интернет плюс
[76]	0,04%	Программа ГИА	http://mgau.ru	06 Мая 2020	Модуль поиска Интернет плюс
[77]	0%	Леонтьев, Александр Николаевич Разработка информационной аналитической с...	http://dlib.rsl.ru	05 Авг 2019	Коллекция РГБ
[78]	0,09%	https://linguanet.ru/upload/medialibrary/f73/f730c7582762a5521f593f6f8f66cbe8.pdf	https://linguanet.ru	25 Авг 2017	Модуль поиска Интернет плюс
[79]	0%	JIT-компиляция — Википедия	https://ru.wikipedia.org	14 Фев 2019	Модуль поиска Интернет плюс
[80]	0%	Машинный код и байт код: на каком языке говорит ваша программа?	https://javarush.ru	10 Янв 2021	Модуль поиска Интернет плюс
[81]	0%	https://linguanet.ru/upload/medialibrary/f73/f730c7582762a5521f593f6f8f66cbe8.pdf	https://linguanet.ru	25 Авг 2017	Модуль поиска Интернет
[82]	0%	JIT-компиляция — Википедия	https://ru.wikipedia.org	14 Фев 2019	Модуль поиска Интернет
[83]	0%	Базовые технологии разработки сайтов и сопутствующего программного обеспе...	https://moscowwebstudio.ru	13 Окт 2020	Модуль поиска Интернет плюс
[84]	0%	Автоматизированная система составления расписания в школе	https://revolution.allbest.ru	14 Дек 2020	Модуль поиска Интернет плюс
[85]	0%	https://asu.tusur.ru/learning/010402/d08/010402-d08-lecture.pdf	https://asu.tusur.ru	14 Дек 2020	Модуль поиска Интернет плюс
[86]	0%	Проект учащейся 8 класса "Сравнение операционных систем Android от Google и I...	https://nsportal.ru	10 Янв 2021	Модуль поиска Интернет плюс
[87]	0%	The-eBook. Книга об электронных книгах. Основы, контент, устройства, программ...	http://dlib.rsl.ru	05 Авг 2019	Коллекция РГБ
[88]	0,08%	http://vital.lib.tsu.ru/vital/access/services/Download/vital:11548/SOURCE01	http://vital.lib.tsu.ru	07 Сен 2020	Модуль поиска Интернет плюс
[89]	0,01%	Научный совет по сравнительной педагогике при Отделении философии образов...	http://instrao.ru	23 Июл 2020	Модуль поиска Интернет плюс
[90]	0%	Новости	https://mordgpi.ru	17 Дек 2020	Модуль поиска Интернет плюс
[91]	0%	РАЗРАБОТКА ИНФОРМАЦИОННОГО ПОРТАЛА «ИЗУЧЕНИЕ ИНОСТРАННОГО ЯЗЫ...	https://scienceforum.ru	10 Янв 2021	Модуль поиска Интернет плюс
[92]	0%	Java библиотеки и фреймворки	http://java-online.ru	10 Янв 2021	Модуль поиска Интернет плюс
[93]	0%	Java библиотеки и фреймворки	http://java-online.ru	10 Янв 2021	Модуль поиска Интернет плюс
[94]	0%	https://books.ifmo.ru/file/pdf/2496.pdf	https://books.ifmo.ru	12 Фев 2020	Модуль поиска Интернет плюс
[95]	0%	https://books.ifmo.ru/file/pdf/2496.pdf	https://books.ifmo.ru	10 Янв 2021	Модуль поиска Интернет плюс
[96]	0,02%	Скачать	http://worldreferat.ru	13 Мая 2020	Модуль поиска Интернет плюс

[97]	0%	Введение, Описание языка программирования Java, Общие характеристики Java - ...	https://studbooks.net	10 Янв 2021	Модуль поиска Интернет плюс
[98]	0%	Достоинства и недостатки Xamarin / Блог компании Microsoft / Хабр	https://habr.com	15 Мая 2020	Модуль поиска Интернет плюс
[99]	0%	Достоинства и недостатки Xamarin / Блог компании Microsoft / Хабр	https://habr.com	10 Янв 2021	Модуль поиска Интернет плюс

Министерство науки и высшего образования Российской Федерации 88
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ЯДЕРНЫЙ УНИВЕРСИТЕТ «МИФИ» 16
ИНСТИТУТ ИНТЕЛЛЕКТУАЛЬНЫХ КИБЕРНЕТИЧЕСКИХ СИСТЕМ 89

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И ТЕХНОЛОГИИ

На правах рукописи

УДК 004.457

ЛАРИН МИХАИЛ ОЛЕГОВИЧ

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ ПРОВЕДЕНИЯ
ЗАНЯТИЙ ДЛЯ ЛЮДЕЙ С ОГРАНИЧЕННЫМИ ВОЗМОЖНОСТЯМИ
СЛУХА 16

Выпускная квалификационная работа бакалавра

Направление подготовки 09.03.01 Информатика и вычислительная техника 16

Выпускная квалификационная
работа защищена 76

«__» _____ 2020 г.

Оценка _____

Секретарь ГЭК _____

г. Москва

2021

Аннотация

Содержание пояснительной записки: 88 страниц, 41 рисунок, 6 таблиц, 10 используемый источник литературы

Ключевые слова: ОВС, МОБИЛЬНОЕ ПРИЛОЖЕНИЕ, КЛИЕНТ, СЕРВЕР.

Объектом исследования является повышение эффективности проведения занятий для людей с ограниченными возможностями слуха.

Предмет исследования — проведение занятий для людей с ограниченными возможностями слуха.

Целью работы является разработка и реализация мобильного приложения для проведения занятий для людей с ограниченными возможностями слуха.

В ходе работы над ВКР был проведен исследование предметной области, установлены задачи и области применения реализуемого проекта. При проведении исследования сделан обзор возможных аналогов, рассмотрены приложения, предлагающие тот же функционал по распознаванию речи, что и разрабатываемая система.

Для определения методов реализации проекта поэтапно рассмотрены требования, функциональность, возможная архитектура.

В результате проведенного исследования разработана и реализовано мобильное приложение для проведения занятий для людей с ограниченными возможностями слуха, которое позволяет визуально воспринимать речь.

Добавлено примечание ([ML1]): Что тут вообще указывать то, господь-господь?

Добавлено примечание ([ML2]): Разобраться, о чем тут должно быть вообще

Оглавление

АННОТАЦИЯ.....	1
ВВЕДЕНИЕ	6
Термины и сокращения.....	7
1 Обзорная часть.....	8
1.1 Обзор технологий для распознавания голоса	8
1.2 Анализ аналогов приложения.....	17
1.3 Обзор технологий для коррекции текста	19
1.4 Обзор мобильных операционных систем.....	20
2 Расчетно-конструкторская часть.....	21
2.1 Определение требований к приложению	21
2.2 Выбор базы данных.....	23
2.3 Разработка структуры базы данных.....	24
2.4 Выбор языка и платформы разработки	27
2.5 Разработка архитектуры приложения.....	29
2.6 Разработка структуры интерфейса взаимодействия пользователя с системой	34
2.6.1 Алгоритм авторизации пользователя при загрузке приложения	39
2.6.2 Алгоритм регистрации пользователя	40
2.6.3 Алгоритм редактирования данных пользователя.....	41
2.6.4 Алгоритм создания комнаты (группы).....	42
2.6.5 Алгоритм отображения списка доступных комнат (групп).....	43
2.6.6 Алгоритм отображения информации о комнате (группе).....	44
2.6.7 Алгоритм редактирования комнаты (группы).....	45
2.6.8 Алгоритм подключения к комнате (группе).....	46

2.6.9 Алгоритм удаления комнаты (группы)	47
2.6.10 Алгоритм отправки сообщений	48
2.6.11 Алгоритм обработки входящих сообщений	49
2.6.12 Алгоритм распознавания голоса в текст и отправка распознанного текста пользователем с ролью «Ведущий»	50
2.7 Разработка плана проведения тестирования и отладки	51
3 Экспериментальная часть	52
3.1 Реализация базы данных	52
3.2 Реализация разработанных алгоритмов	54
3.2.1 Алгоритм авторизации пользователя при загрузке приложения	54
3.2.2 Алгоритм регистрации пользователя	57
3.2.3 Алгоритм редактирования данных пользователя.....	59
3.2.4 Алгоритм создания комнаты (группы).....	61
3.2.5 Алгоритм отображения списка доступных комнат (групп).....	63
3.2.6 Алгоритм отображения информации о комнате (группе).....	65
3.2.7 Алгоритм редактирования комнаты (группы).....	67
3.2.8 Алгоритм подключения к комнате (группе).....	69
3.2.9 Алгоритм удаления комнаты (группы)	71
3.2.10 Алгоритм отправки сообщений	73
3.2.11 Алгоритм обработки входящих сообщений	76
3.2.12 Алгоритм распознавания голоса в текст и отправка распознанного текста пользователем с ролью «Ведущий»	78
3.3 Тестирование и отладка	80
3.3.1 Тестирование алгоритма авторизации пользователя.....	80

3.3.2 Тестирование алгоритма регистрации пользователя.....	80
3.3.3 Тестирование алгоритма редактирования данных пользователя	81
3.3.4 Тестирование алгоритма создания комнаты (группы)	81
3.3.5 Тестирование алгоритма отображения списка доступных комнат (групп).....	82
3.3.6 Тестирование алгоритма отображения информации о комнате (группе).....	82
3.3.7 Тестирование алгоритма редактирования комнаты (группы)...	83
3.3.8 Тестирование алгоритма подключения к комнате (группе).....	83
3.3.9 Тестирование алгоритма удаления комнаты (группы)	84
3.3.10 Тестирование алгоритма отправки сообщений	84
3.3.11 Тестирование алгоритма обработки входящих сообщений	85
3.3.12 Тестирование алгоритма распознавания голоса в текст и отправка распознанного текста пользователем с ролью «Ведущий».....	85
3.5 Руководство пользователя	89
Заключение.....	87
Список использованной литературы	88

Введение

Главная цель технологического развития всегда состояла в упрощении жизни человека. В наше время существует огромное количество технологий и методик для помощи людям с ограниченными возможностями. Технологии позволяют частично восстановить потерянные конечности или утраченный функционал. Но даже сейчас существуют сферы, в которых люди с ограниченными возможностями чувствуют себя некомфортно. Одна из таких сфер – образование. Несмотря на наличие специальных центров для людей с ограниченными возможностями и специальных методик обучения. Все равно есть образовательные программы, которые были бы интересны таким людям, но из-за ограничений они не могут ими воспользоваться.

Как пример: люди с ограниченными возможностями слуха. Это люди, у которых частично снижена способность обнаруживать и понимать звуки. Иногда такие люди хотят освоить программу, которой нет в специальном образовательном учреждении. Особенно если речь идет о каком-то дополнительном курсе. И также существуют ситуации, когда подобные дети учатся в обычных образовательных учреждениях. И им остается либо полагаться на устройства повышающие способность понимать и обнаруживать звуки. Либо на визуальное восприятие информации. Очевидно, что не всегда презентация содержит достаточно информации, так как прежде всего она служит для иллюстрации того, что рассказывается.

Цель данного проекта заключается в том, чтобы, используя имеющиеся технологии позволить людям с ограниченными возможностями слуха как можно эффективно воспринимать информацию, а также повысить качество коммуникации между ведущим курса и участниками курса с ограниченными возможностями слуха. Результатом должно стать мобильное приложение, предоставляющее функционал трансляции речи ведущего на мобильные устройства участников, переводя речь ведущего в информацию, которую можно воспринимать визуально.

Термины и сокращения

OBC – ограниченные возможности слуха

Группа, комната – виртуальное объединение пользователей для согласованного взаимодействия между собой.

HTTP – протокол прикладного уровня использующийся для передачи данных. Включает в себя несколько методов, например: GET и POST. Методы отличаются типом способом передачи информации и сферой применения. GET-запросы часто используются для получения необходимой информации, данные для таких запросов передаются в строке запроса. POST-запросы чаще используются для отправки данных. Данные в таком запросе передаются в отдельном теле запроса, формат которого может различаться.

URL – система унифицированных адресов электронных ресурсов, используется для обозначение электронного адреса.

Клиент – приложение, содержащее пользовательский интерфейс. Предназначено для подготовки данных клиента перед отправкой на сервер и в обратном процессе подготовки данных с сервера перед отображением клиенту.

Сервер – приложение, выполняющее основной функционал. Все основные алгоритмы выносятся на серверную часть для обеспечения централизованной обработки запросов клиентов.

1 Обзорная часть

Разрабатываемое мобильное приложение должно иметь функционал по распознаванию и преобразованию в текст голоса ведущего с последующей трансляцией распознанного текста на подключенные устройства участников. Также пользователи должны иметь возможность создавать и подключаться группам для объединения некоторого кол-ва участников для согласованного взаимодействия между собой в рамках группы.

1.1 Обзор технологий для распознавания голоса

Распознавание речи – процесс автоматического преобразования речевого сигнала в цифровую информацию, например текст. Изначально использовались для помощи людям с ограниченными возможностями. В настоящее время данные технологии активно используются в повседневной жизни и сфере бизнеса.

Рассмотрим наиболее часто используемые технологии для распознавания голоса.

Yandex SpeechKit

Yandex SpeechKit – технология для синтеза и распознавания речи разработана российской компанией Yandex (Яндекс). Компания с 2012 г. занимается разработками в области распознавания естественной речи. Данная технология впервые была представлена в 2013 г. на ежегодной технологической конференции Яндекса. Технология поддерживает такие платформы как: Android, IOS и windows phone. Также технология активно используется в рест-серверных Web-приложениях. Технология активно используется в проектах компании Яндекс, таких как: голосовой помощник Алиса, Яндекс.Карты и др [1].

Технология работает по принципу рест-сервиса. Это значит, что для интеграции с Yandex SpeechKit необходимо проводить аутентификацию по токену. И использовать HTTP методы POST и GET для получения результата

распознавания или синтеза речи. Само распознавание или синтез речи проходит на серверах Яндекса. Следовательно качество работы сервиса напрямую зависит от качества интернет-соединения и загруженности сервера [1].

Распознавание аудио происходит в несколько этапов:

1. Из полученного текста выделяются слова. Обычно существует несколько вариантов распознанного слова.
2. Варианты распознанного слова проверяются с помощью языковой модели. Модель проверяет, насколько согласуется новое слово со словами, распознанными ранее. Для каждого варианта указывается параметр доверия, отражающий насколько распознанное слово, подходит под контекст.
3. Распознанный текст Обработывается – числительные преобразуются в цифры, расставляются необходимые знаки препинания (например, дефисы) и т. д. Затем преобразованный текст отправляется в теле ответа.

Достоинства:

- Технология Yandex SpeechKit учитывает стилистические и лексические особенности устной речи, а также вероятности сочетания слов для повышения точности распознаваемого текста за счет контекста.
- Три возможных режима распознавания голоса:
 - Короткие аудио – используется для распознавания речи из короткого, до 1 минуты, одноканального аудио.
 - Длинное аудио – используется для распознавания речи из многоканального аудио, не ограниченной длины. Длина аудио напрямую влияет на скорость распознавания речи.

- Потокное аудио – позволяет в рамках одного соединения отправлять аудиофрагменты и получать результаты. Используется для получения промежуточных результатов в реальном времени.
- Поддержка трех языков: русский, английский, турецкий
- Премиум голоса для синтеза речи – набор различных голосов доступных для синтеза речи. Технология оценивает текст и подбирает интонации на основе контекста, характерные для человека
- Поддержка различных форматов аудио

Недостатки:

- Так как распознавание или синтез речи происходит на серверах Яндекса, то качество и скорость работы сервиса напрямую зависят от качества интернет-соединения и загруженности серверов Яндекса.
- Невозможно проводить распознавание или синтез сразу на устройстве, что само по себе увеличивает время, между тем как была сказана фраза и был получен результат распознавания, так как нам необходимо ждать ответ от сервера.
- Поддерживается только три языка: Русский, Английский, Турецкий
- Сервис является платным. Ежемесячный тариф рассчитывается на основе типа и суммарной длительности аудио дорожек в месяц. Следовательно, необходимо детально просчитывать экономическую выгоду от использования этой технологии.
- Качество голосов для синтеза речи хуже, чем у конкурентов.

На основании вышеперечисленного, технология Yandex SpeechKit выглядит крайне интересно. Она имеет достаточно большое количество весомых достоинств. Наиболее интересным вариантом распознавания речи является потоковое распознавание речи. Этот режим позволяет нам получать промежуточные результаты распознавания, не дожидаясь конца фразы или предложения. Данный формат как раз необходим и является ключевой

особенностью, так как при проведении лекций или занятий необходимо распознавать текст в реальном времени небольшими отрывками, а не огромными кусками, чтобы участники могли быть в «потоке».

Ключевыми же недостатками является рест-сервисная архитектура и наличие абонентской платы. Рест-сервисная архитектура на несколько секунд увеличивает суммарное время распознавания речи. При хорошем качестве интернет-соединения и приемлемой нагрузке на серверах Яндекса скорость отклика будет удовлетворительной, но если один из этих аспектов ухудшится, то, следовательно, увеличится время отклика. В свою очередь время отклика является критичным. Также этот метод распознавания имеет ряд ограничений:

- Время между отправкой фрагментов аудио должно примерно совпадать с длительностью фрагментов. Нельзя отправлять сообщения слишком редко или слишком часто. Например, каждые 400мс необходимо отправлять аудио фрагмент длительностью 400мс. Если пауза между отправкой будет больше 5 секунд, текущее соединение будет закрыто, и нужно будет открывать его заново.
- Максимальная длительность аудио в рамках одной сессии – 5 минут. Затем сессию необходимо открывать повторно.
- Размер переданных аудиоданных не должен превышать 10 мегабайт.

Вторым недостатком является наличие абонентской платы. Абонентская плата напрямую зависит от длительности обрабатываемых аудиоданных. Если рассматривать ситуацию, когда проводятся лекции 22 дня в месяц по 8 час. в день, ежемесячная плата находится в районе 5 тыс рублей. Цифры примерные, но они отражают необходимость точного расчёта экономической целесообразности использования данной технологии. В остальном же технология хорошо зарекомендовала себя в других сервисах от компании Яндекс.

Google Cloud Speech API

Google Cloud Speech API – технология разработана компанией Google. Изначально технология была встроена в платформу Android и использовалась для поиска голосом, и диктовки. Это позволяло разработчикам использовать данный функционал в своих приложениях. Но примерно в 2013-2014 годах компания Google закрыла доступ к этому сервису так как сильно увеличилась нагрузка на сервера Google. Позднее компания представила сервис Google Cloud Speech API который по факту был описанной и регламентированной версией предыдущей технологии [2].

Принцип работы схож с Yandex SpeechKit. Это тоже рест-сервис, с которым обмен информацией происходит посредством HTTP запросов. Сервис так же обладает рядом преимуществ, у сервиса явно выше качество распознавания текста, более высокого качества голоса для синтеза речи. Больше количество поддерживаемых языков. Но у сервиса существует один ключевой недостаток:

- Значительно более высокая цена по отношению к Yandex SpeechKit. Каждые 15 секунд распознавания аудио стоят примерно 0,006 USD. Если рассматривать сценарий, когда проводятся лекции 22 дня в месяц по 8 часов в день, итоговая сумма выходит больше 12 тыс. рублей за одно только устройство. Крайне высокая цена, использование данной технологии экономически не выгодно.

Android Speech API (Google Speech API)

Android Speech API – технология для распознавания и синтеза речи разработанная компанией Google. Изначально разрабатывалась под платформу Android, была встроена и активно использовалась внутри приложений с возможностью голосового поиска или диктовки. На данный

момент также используется в голосовом помощнике от Google. В дальнейшем на ее основе была создана технология Google Cloud Speech API. У Android Speech API есть огромное и основное отличие от Google Cloud Speech API, эта технология является интегрированной в платформу Android. Следовательно, распознавать и синтезировать речь можно сразу на устройстве, не будучи привязанным к качеству интернет-соединения или серверам, на которых происходит обработка. На данный момент эта технология активно используется во многих приложениях на платформе Android и крайне хорошо себя зарекомендовала [3].

Достоинства:

- Технология интегрирована в платформу Android, что позволяет нам распознавать и синтезировать речь прямо на устройстве. Такой подход уменьшает время отклика и исключает из алгоритма возможные проблемы с интернет-соединением и загрузкой серверов.
- Технология бесплатна. Все что необходимо для ее использования это непосредственно само устройство на платформе Android.
- Поддерживается огромное количество языков. Технология может распознавать и синтезировать речь практически на всех известных языках.
- Высокое качество голоса для синтеза речи.

Недостатки:

- Нет потокового распознавания речи. Существует только один формат обработки аудио для распознавания речи, и его особенность в том что он распознает только полную аудио дорожку, которая получается либо при длительной паузе в речи либо при ручной остановке алгоритма.
- Поддерживается только одна платформа: Android. На других платформах таких как IOS или Windows Phone эта технология не

поддерживается. Но на этих платформах существуют свои аналоги от компаний Apple и Microsoft.

Основным недостатком данной технологии является отсутствие потокового распознавания речи. Для примера необходимо рассмотреть алгоритм распознавания:

1. Настраивается конфигурация для распознавания. Указываются необходимы лингвистически словари, указывается периоды и промежутки, настраивается устройство ввода.
2. Запускается алгоритм распознавания речи.
3. Алгоритм формирует аудио дорожку на основе поступающих данных с устройства ввода
4. Если речь прервалась, или пауза речи выше указанной, то алгоритм прекращает запись звука и формирования аудио дорожки и приступает к распознаванию.
5. Распознанный текст выдается в результат.

Данный алгоритм не подразумевает получение промежуточных результатов распознавания. В лучшем случае удастся получить результат по предложениям, но если человек с высоким темпом речи, то результат мы получим сильно позже. Если рассматривать процесс ведения лекций или семинаров, то такой вариант не подходит, так как необходимо получать результат в реальном времени что бы участники могли реагировать на события также в реальном времени. При использовании данной технологии проблему с отсутствием потокового распознавания речи можно решить с помощью периодической остановки алгоритма. Так если мы с определенной периодичностью на шаге 4 алгоритма будем вызывать прерывание алгоритма, то мы циклически будем проходить пункты 2-5, тем самым фактически получая потоковое распознавание речи. Но у такого подхода есть ряд недостатков. Основной заключается в том, что слова распознаются и корректируются на основе контекста фразы. А при прерывании алгоритма

контекст фактически прерывается и возникают ситуации, когда распознанные слова находятся не в том падеже или теряют окончания. Также могут быть потеряны союзы. Эту проблему можно решить, используя инструменты коррекции текста, когда будут анализироваться распознанный текст из моментов времени t и $t+1$ и на основе контекста этих двух составляющих корректировать текст.

В целом технология Android Speech API является лучшим вариантом, так как интегрирована в платформу и не зависит от качества сетевого соединения. Но корректная работа будет обеспечена только в том случае если будет решена проблема с ухудшением качества распознанного текста из-за потери контекста в потоковом распознавании.

Другие технологии

Существует некоторое количество библиотек с открытым исходным кодом вроде Pocketsphinx и других, но они не могут рассматриваться в качестве потенциальных технологий так как работают по принципу реагирования на конкретную заранее заготовленную фразу.

Вывод

Из перечисленных выше технологий можно выделить две:

Yandex SpeechKit

Достоинства:

- Наличие потокового распознавание речи
- Высокое качество распознавания
- Относительная простота интеграции
- Возможна поддержка различных мобильных платформ

Недостатки:

- Зависимость от интернет-соединения и нагрузки на сервера сервиса
- Наличие весомой абонентской платы

Android Speech API

Достоинства:

- Технология интегрирована в платформу Android, что в свою очередь убирает влияние качества интернет-соединения и загрузке серверов сервиса на скорость отклика при распознавании речи
- Не имеет абонентской платы.

Недостатки:

- Повышенная сложность реализации. Так как изначально технология не поддерживает потоковое распознавание речи. Алгоритм распознавания требует доработки и постобработки полученного результата с помощью коррекционных инструментов.
- Поддерживает только платформу Android

Остальные технологии либо не удовлетворяют условиям, либо слишком дороги.

1.2 Анализ аналогов приложения

16

На данный момент не существует приложений, нацеленных конкретно на повышение эффективности образовательного процесса для людей с ограниченными возможностями слуха. Рассмотрим несколько приложений с функционалом перевода речи собеседника в текст с последующим отображением на экране.

16

Яндекс.Разговор

Яндекс.Разговор – приложение, разработанное компанией Яндекс, для помощи людям с ограниченными возможностями слуха. Приложение переводит речь в текст и обратно. Придумано и разработано группой студентов из Московского физико-технического института с использованием технологии распознавания речи Yandex SpeechKit [4].

Все что говорит один из собеседников переводится в речь и отображается на экране собеседника. Также есть возможность произносить вслух напечатанные слова.

Достоинства:

- Разрабатывается и поддерживается компанией Yandex
- Удобный интерфейс и хорошим дизайном
- Использует технологию Yandex SpeechKit
- Есть история диалогов
- Приложение бесплатное

Недостатки:

- Одновременно можно вести диалог только с одним человеком
- Точность распознавания зависит от скорости речи и дикции собеседника. Выше скорость речи – ниже точность.
- Поддерживается только один язык: Русский
- Нет возможности экспортировать диалог

- Судя по отзывам большое количество ошибок.

RogerVoice

RogerVoice – приложение разработанное компанией Rogervoice. Приложение предоставляет функционал с помощью, которого люди с ограниченными возможностями слуха могут разговаривать по телефону. Приложение использует собственную технологию распознавания речи, для перевода голоса собеседника в текст и отображения полученного текста на экране [5].

Достоинства:

- Собственная технология распознавания голоса
- Удобный интерфейс с отличным дизайном интерфейса
- Бесплатное приложение

Недостатки:

- Не удалось найти информацию по поддерживаемым языкам
- Только один собеседник одновременно
- Нет локализации на русский язык
- Судя по отзывам большое количество критичных ошибок

Вывод

Рассмотренные приложения невозможно использовать в образовательном процессе, так как каждое из них одновременно может работать только с одним собеседником. Такая схема может использоваться только при индивидуальном репетиторстве, но не при массовых занятиях. Также приложение RogerVoice невозможно использовать, так как у него нет локализации на русский язык. В целом становится понятно, что на данный момент нет приложений, которые могли бы использоваться при обучении людей с ограниченными возможностями слуха.

1.3 Обзор технологий для коррекции текста

Алгоритмы распознавания текста имеют крайне высокую точность, но иногда результат все равно содержит ошибки, особенно если появляются сильные внешние шумы либо помехи. Для коррекции подобных ошибок можно использовать инструменты для коррекции текста. Ниже приведены некоторые из актуальных инструментов для коррекции.

LanguageTool

LanguageTool – библиотека разработана немецкой компанией LanguageTooler GmbH в 2016 году. Интегрирована в библиотеку текстового поиска Lucene и поставляется вместе с ней.

Технология проста в интеграции и поддерживает большое количество языков. Распространяется по лицензии компании Apache, стоимость определяется от количество проверяемых ежемесячно слов. Библиотека является встраиваемой, что позволяет анализировать и исправлять текст непосредственно в приложении [6].

Яндекс. Спеллер

Яндекс. Спеллер – сервис, разработанный компанией Яндекс. Является рест-сервисом следовательно запросы к сервису и получение ответа осуществляется через HTTP запросы методами. Ключевыми недостатками сервиса является ограничение в 10 миллионов символов в сутки и зависимость скорости отклика от качества интернет-соединения и нагрузки на серверах сервиса [7].

1.4 Обзор мобильных операционных систем

Android – изначально разрабатывалась компанией Android inc. которую затем выкупила компания Google. Релиз операционной системы состоялся 23 сентября 2008 года. Операционная система основана на ядре Linux и собственной реализации виртуальной машины Java от компании Google. Система получила огромное распространение и большее количество смартфонов на данный момент построены на операционной системе Android [8].

IOS – операционная система была выпущена в 2007 году компанией Apple. IOS основана на ядре XNU и языке C, C++. Операционная система используется в основном на устройствах компании Apple, из-за чего популярность меньше, чем у операционной системы Android [9].

2 Расчетно-конструкторская часть

В данной главе описываются основные требования к приложению, обоснование выбора базы данных и ее схема для разработки, а также выбор языка и платформы для разработки. Представлена и описана архитектура приложения и описаны необходимые алгоритмы с приложенными блок-схемами алгоритмов.

2.1 Определение требований к приложению

- Мобильное приложение должно разрабатываться на платформе Android и поддерживать версии выше Android 7.
- Приложение должно поддерживать ввод информации через микрофон устройства, или внешний микрофон.
- Приложение должно иметь функционал вывода звука через динамики устройства или внешние устройства воспроизведения звука.
- Приложение должно иметь функционал распознавания голоса пользователя в текст с последующей отправкой распознанного текста на устройства подключенных пользователей.
- В приложении должен быть функционал создания групп для объединения пользователей и подключения к созданной группе. Распознанный текст должен передаваться на все устройства, подключенные к одной группе.
- Приложение должно иметь клиент-серверную архитектуру.
- Клиентская часть приложения должна иметь удобный для пользователя интерфейс, функционал регистрации и аутентификации пользователей, а также функционал по созданию групп и подключения к созданным группам, как и функционал отправки и получения сообщений в рамках отдельно взятой группы. Распознавание голоса должно происходить на клиентской части приложения. Затем распознанный текст должен отправляться на серверную часть приложения. Распознавание речи должно происходить в реальном времени. Также клиентская часть

приложения должна содержать функционал по синтезу речи из текста, для воспроизведения написанного текста голосом.

- Серверная часть приложения должна иметь функционал регистрации и аутентификации пользователей, передачи сообщений, отправленных в рамках группы, функционал создания группы и подключения к ранее созданным группам. Сообщения пользователей в рамках группы должно передаваться с использованием протокола WebSocket, для обеспечения одновременной доставки сообщений на все, подключённые к группе устройства, в реальном времени.
- Должна быть реализована реляционная база данных для хранения зарегистрированных пользователей, групп и др.

2.2 Выбор базы данных

В качестве базы данных при разработке была выбрана Oracle Database – объектно-реляционная СУБД от компании Oracle.

Oracle Database был выбран т.к. имеет ряд преимуществ перед Postgres или MySQL таких как:

- Все изменения БД сначала пишутся в кэш, а потом асинхронно пишутся на диск. Вместе с этим изменения пишутся в специальный лог, который позволяет в случае сбоя восстановить часть данных.
- Наиболее эффективное использование индексов: возможность использовать битовые индексы и создавать индексы по функции если это необходимо.
- Наиболее эффективное потребление памяти. Например: пустые строки никогда не хранятся в таблицах. При вставке они заменяются значениями NULL, что позволяет экономить память.
- Возможность быстро развернуть кластерный доступ к базе данных.
- Самым важным преимуществом является расширенная поддержка компанией Oracle и высокая популярность у разработчиков. Oracle Database используется в огромном количестве продуктов, в следствие чего поддерживать ее гораздо проще так как специалистов, погруженных в архитектуру Oracle Database, на рынке очень много.

Помимо этих преимуществ Oracle Database имеет огромное кол-во архитектурных решений, отличающих ее от конкурентов. Но в рамках разрабатываемого проекта они рассматриваются.

2.3 Разработка структуры базы данных

База данных Oracle содержит данные о зарегистрированных клиентах и созданных группах, которые здесь называются комнатами.

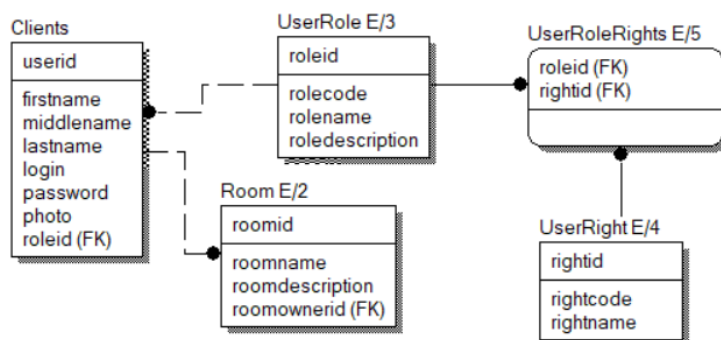


Рисунок 2.1 Структура БД.

Таблица 1. Таблица clients содержащая зарегистрированных клиентов.

clients		
Имя поля	Тип	Описание
userid	number	Числовой идентификатор пользователя. Является первичным ключом.
firstname	varchar2 (255 byte)	Имя пользователя
middlename	varchar2 (255 byte)	Отчество пользователя
lastname	varchar2 (255 byte)	Фамилия пользователя
login	varchar2 (18 byte)	Логин пользователя. Является уникальным полем.
password	varchar2 (18 byte)	Пароль пользователя.
photo	clob	Фото пользователя в кодировке Base64.

Таблица 2. Таблица room содержащая комнаты (группы)

room		
Имя поля	Тип	Описание
roomid	number	Числовой идентификатор комнаты. Является первичным ключом.
roomname	varchar2 (50 byte)	Имя комнаты. Является уникальным значением
roomdescription	varchar2 (255 byte)	Текстовое описание комнаты.
roomownerid	number	Идентификатор пользователя, который создал комнату. Является внешним ключом для связи с таблицей CLIENTS

Таблица 3. Таблица userrole содержащая сущность роли пользователей

userrole		
Имя поля	Тип	Описание
roleid	number	Идентификатор роли пользователя, является первичным ключом.
rolecode	varchar2 (255 byte)	Кодовый идентификатор роли пользователя
rolename	varchar2 (255 byte)	Имя роли пользователя
roledescription	varchar2 (255 byte)	Описание роли пользователя

Таблица 4. Таблица userright содержащая сущность права пользователей

userright		
Имя поля	Тип	Описание
rightid	number	Числовой идентификатор права. Является первичным ключом
rightcode	varchar2 (255 byte)	Символьный идентификатор права
rightname	varchar2 (255 byte)	Имя права

Таблица 5. Таблица userrolerights содержащая роли и права пользователей

userrolerights		
Имя поля	Тип	Описание
roleid	number	Идентификатор роли пользователя, является первичным ключом.
rightid	number	Числовой идентификатор права. Является первичным ключом

2.4 Выбор языка и платформы разработки

В качестве платформы разработки для мобильного приложения была выбрана платформа Android. Платформа была выбрана из-за большей популярности на рынке, устройств на платформе Android на данный момент больше, чем на платформе IOS, Windows Phone или других. Следовательно потенциальная аудитория больше, чем на других платформах.

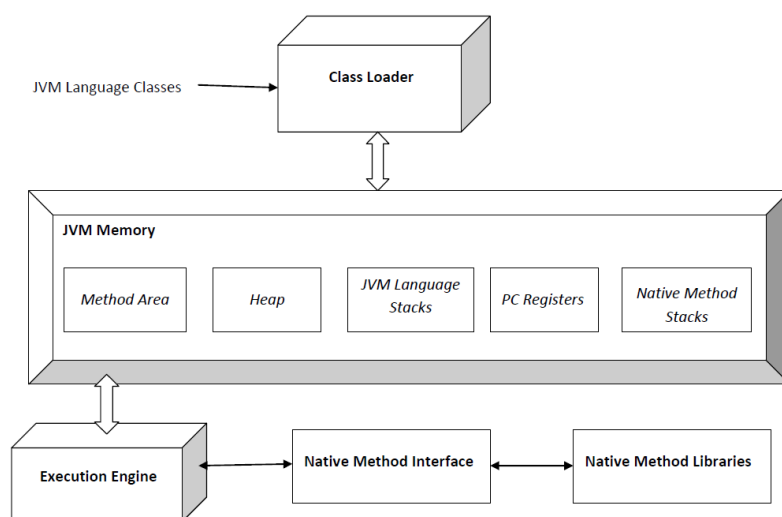
Так как платформа Android основана на языке Java от компании Google, то в качестве языка разработки был выбран язык Java. Java - объектно-ориентированный язык строго типизированный программирования разработанный компанией Sun Microsystems которую в дальнейшем приобрела компания Oracle. Отличительной особенностью языка Java является то, что приложения обычно транслируются в специальный байт-код в результате чего они могут работать на любой компьютерной архитектуре, для которой существует реализация виртуальной Java-машины.

Виртуальная Java-машина исполняет байт-код java, который был создан из исходного текста программы, написанной на языке Java с помощью компилятора Java. Виртуальная машина Java может также использоваться для выполнения программ, написанных на других языках программирования. Исходный код некоторых программ может быть скомпилирован в байт-код Java который затем может быть выполнен с помощью виртуальной машины Java. Так как виртуальные машины Java доступны на многих аппаратных и программных платформах, то это позволяет использовать один скомпилированный байт-код на многих платформах. Виртуальные машины Java обычно содержат интерпретатор байт-кода, однако, для повышения производительности во многих машинах также применяется JIT-компиляция часто исполняемых фрагментов байт-кода в машинный код [10].

JIT-компиляция – технология использующая компиляцию байт-кода в машинный код или в другой формат непосредственно во время работы

78

программы, что позволяет увеличить производительность программных систем.



19

Рисунок 2.2 Обзор архитектуры JVM на базе версии Java SE 7 [10].

2.5 Разработка архитектуры приложения

Приложение должно иметь клиент-серверную архитектуру. Приложение делится на два компонента: клиент и сервер. Схематическая архитектура представлена ниже.

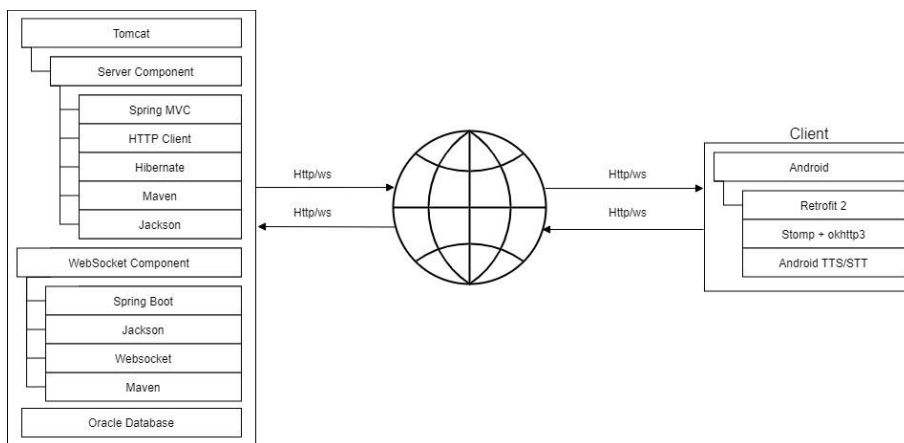


Рисунок 2.3 Архитектура приложения.

Клиент

Клиентская часть является мобильным приложением на платформе Android. Для построения интерфейса пользователя используются стандартные Android компоненты. Также для реализации требуемого функционала используется несколько ключевых библиотек:

- Retrofit 2 – HTTP-клиент для Android и Java. Библиотека разработана компанией Square и предназначена для отправки и обработки HTTP запросов. Retrofit позволяет сделать полноценный REST-клиент, который может выполнять POST, GET, PUT, DELETE. Для обозначения типа и других аспектов запроса используются Java-аннотации.
- Комбинация библиотек Stomp+okhttp3. Используется для реализации протокола WebSocket, для обмена сообщениями между клиентами в реальном времени.

- Android TTS/STT – библиотека разработанная компанией Google и предназначенная для распознавания речи в текст и обратно, текста в речь.

Сервер

Серверная часть обеспечивает основной функционал приложения, путем обработки запросов с клиентской части. Серверная часть включает в себя несколько компонентов.

Tomcat – контейнер сервлетов для развертки приложений и управления ресурсами. Разрабатывается компанией Apache Software Foundation. Реализует спецификацию сервлетов, спецификацию JavaServer Pages (JSP) и JavaServer Faces (JSF). Разработан на языке Java. Tomcat позволяет запускать веб-приложения и содержит ряд программ для самоконфигурирования. Tomcat по функционалу уступает конкурентам в лице Weblogic или Websphere, но был выбран исключительно из-за простоты первоначальной конфигурации. Архитектура проекта позволяет в дальнейшем использовать любые контейнеры сервлетов. Используется для разворачивания и запуска Server Component.

Server Component – компонент написанный на языке Java. Предназначен для выполнения большей части основного функционала: регистрация и аутентификация пользователей, редактирование данных пользователя, создание групп и удаление групп. Для реализации необходимого функционала компонент использует несколько технологий:

- Spring MVC (Spring Framework) – фреймворк с открытым исходным кодом для платформы Java. Фреймворк обеспечивает архитектуру паттерна Model - View – Controller (Модель – Отображение – Контроллер) при помощи слабо связанных готовых компонентов. Паттерн MVC разделяет аспекты

приложения (логику ввода, бизнес-логику и логику UI), обеспечивая при этом свободную связь между ними.

- HTTP Client – библиотека разработанная компанией Oracle. Используется для отправки и обработки HTTP запросов.
- Hibernate – библиотека разработанная компанией Red Hat для Java. предназначенная для решения задач объектно-реляционного отображения (ORM), самая популярная реализация спецификации JPA. Распространяется свободно на условиях GNU Lesser General Public License. Библиотека решает проблему связи Java-классов и базы данных, а также предоставляет средства для автоматической генерации и обновления набора таблиц. Hibernate автоматизирует генерацию SQL-запросов что позволяет интегрировать практически любые реляционные базы данных в приложение.
- Maven – фреймворк для автоматизации процесса сборки приложения на основе конфигурации описанной на языке POM, являющимся подмножеством XML. На данный момент популярность набирает фреймворк Gradle, построенная на тех-же принципах, но использующий специализированный DSL на Groovy вместо POM-конфигурации что требует погружения в язык программирования Groovy.
- Jackson – библиотека для сериализации Java-объектов в JSON и обратно. Обладает высокой степенью конфигурирования и простотой использования.

WebSocket Component – компонент написанный на языке Java. Предназначенный для обмена сообщениями между пользователями в реальном времени. Реализует работу с протоколом WebSocket – независимый протокол, основанный на протоколе TCP. Он делает возможным более тесное взаимодействие между клиентом и сервером. Для установления WebSocket соединения клиент посылает особый HTTP запрос, с указанием наличия

протокола WebSocket и его типа. После ответа сервера устанавливается соединение и в рамках данного подключения обмен информацией осуществляется в реальном времени. Для реализации функционала обмена сообщениями, протокол WebSocket используется вместе с брокером сообщений. Брокер сообщений – приложение, которое выступает посредником между приложением-источником и приложением-приемником, обеспечивая передачу сообщений по определенному протоколу, в данном случае это протокол WebSocket. Подробнее про брокера сообщений и технологии, используемые в компоненте, ниже.

- Spring Boot – модификация фреймворка Spring Framework. Обладает более обширным функционалом по сравнению Spring Framework. Ключевыми же особенностями являются автоматическая конфигурация и наличие встроенных контейнеров сервлетов, что позволяет не использовать Tomcat или Weblogic как в Server Component, а запускать приложение прямо на машине. Выбран был в первую очередь ради освоения аналога Spring Framework. Также по причине того, что WebSocket Component предназначен для передачи сообщений между клиентами, следовательно конфигурация нужна только для ранее упомянутого брокера сообщений, и Spring Boot, благодаря автоматической конфигурации, позволяет быстро настроить все необходимое для работы. Так же фреймворк имеет встроенный брокер сообщений, который будет использоваться вместе с протоколом WebSocket. При конфигурировании брокера сообщений создаются каналы подключения, называемые также топиками. В дальнейшем если приложение, подключенное к этому каналу, отправляет в него сообщение, это сообщение отправляется на все приложения, которые также подключены к этому каналу. В случае разрабатываемого приложения роль каналов исполняют группы, создаваемые пользователями. При подключении

пользователей к группе, они подписываются на один канал, имеющий идентификатор группы, и таким образом осуществляется обмен сообщениями в реальном времени в рамках группы. Использование фреймворка Spring Boot позволяет в дальнейшем вместо встроенного брокера сообщений интегрировать более функциональные аналоги, например ActiveMQ или Kafka. Они обладают более обширным функционалом по контролю каналов и валидации сообщений, но увеличивают затраты аппаратных и финансовых ресурсов, поэтому был выбран встроенный брокер сообщений от фреймворка Spring Boot.

- Jackson – библиотека для сериализации Java-объектов в JSON и обратно.
- WebSocket – как упоминалось ранее, это независимый протокол, основанный на протоколе TCP. Он делает возможным более тесное взаимодействие между клиентом и сервером. ¹⁷ Используется для реализации функционала обмена сообщениями в реальном времени.
- Maven – фреймворк для автоматизации процесса сборки приложения. Подробно описан в Server Component.

2.6 Разработка структуры интерфейса взаимодействия пользователя с системой

В приложении предусмотрена ролевая модель. На данный момент не предполагается функционал создания и редактирования ролей. Роли создаются разработчиком на этапе настройки системы перед запуском, и предполагается наличие только двух ролей:

- Пользователь – роль по умолчанию для новых пользователей. Роль позволяет искать и подключаться к комнатам (группам), а также участвовать в процессе обмена сообщениями в комнате (группе);
- Ведущий – роль, устанавливаемая для пользователей, которые в дальнейшем собираются проводить лекции или занятия. Роль включает в себя разрешения роли «Пользователь», а также возможность создавать, редактировать и удалять комнаты (группы);

Ниже представлены интерфейсы для авторизации и регистрации пользователя, отображения созданных комнат, для создания комнаты и подключения к комнате по идентификатору. Также интерфейсы для обмена сообщениями для пользователей с ролью «Пользователь» и «Ведущий»;

Авторизация

Логин: _____

Пароль: _____

войти

РЕГИСТРАЦИЯ

Рисунок 2.4 Окно авторизации пользователя

Регистрация

Логин: _____

Имя: _____

Фамилия: _____

Отчество: _____

Пароль: _____

Роль: ☒ Пользователь ☐ Ведущий

ПОДТВЕРДИТЬ

ОТМЕНА

Рисунок 2.5 Окно регистрации пользователя

Подключение к комнате +

Room#12

Room#82

Room#83

Room#95

Room#136

Рисунок 2.6 Окно со списком доступных комнат.

Подключение к комнате +

Room#12

Room#82

Room#83

Room#95

Room#136

Создать комнату

Подключиться

Ред. дан. польз.

Рисунок 2.7 выпадающий список меню на форме доступных комнат.

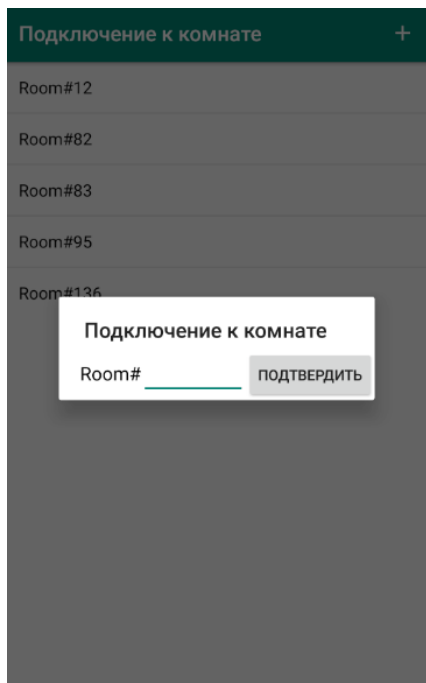


Рисунок 2.8 Подключение к комнате по идентификатору.

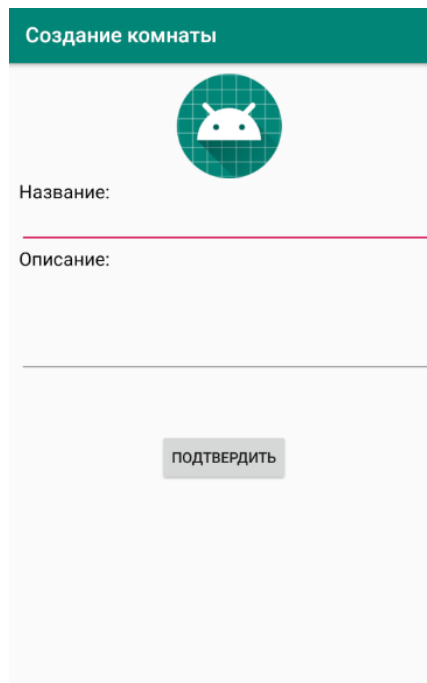


Рисунок 2.9 Окно создания комнаты

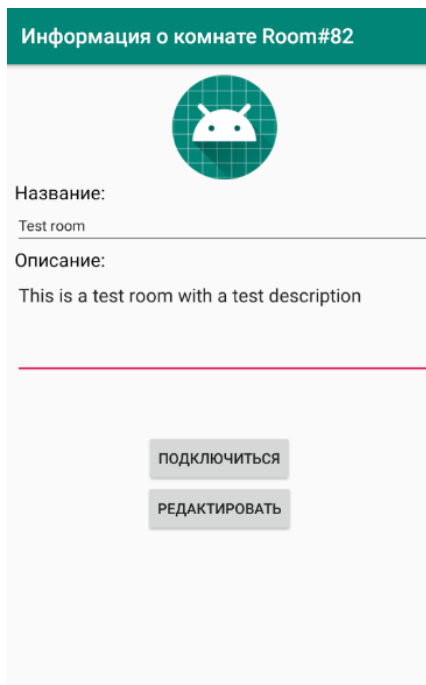


Рисунок 2.10 Окно просмотра информации о комнате.

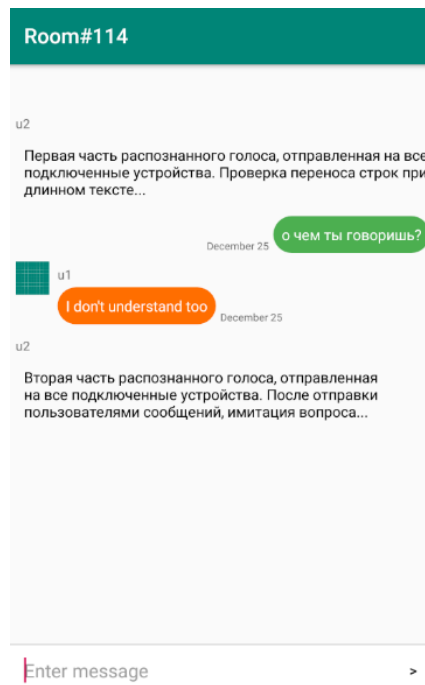


Рисунок 2.11 Окно комнаты. Процесс распознавания голоса и отправки на подключенные устройства, с параллельной обработкой вопросов от пользователей.

Рисунок 2.12 Окно редактирования информации о клиенте.

Рисунок 2.13 Выпадающее меню на форме комнаты с кнопками запуска и остановки механизма распознавания речи.

Ниже приведены блок-схемы и краткое описание каждого алгоритма.

2.6.1 Алгоритм авторизации пользователя при загрузке приложения

Алгоритм инициируется при запуске приложения через иконку на рабочем столе.

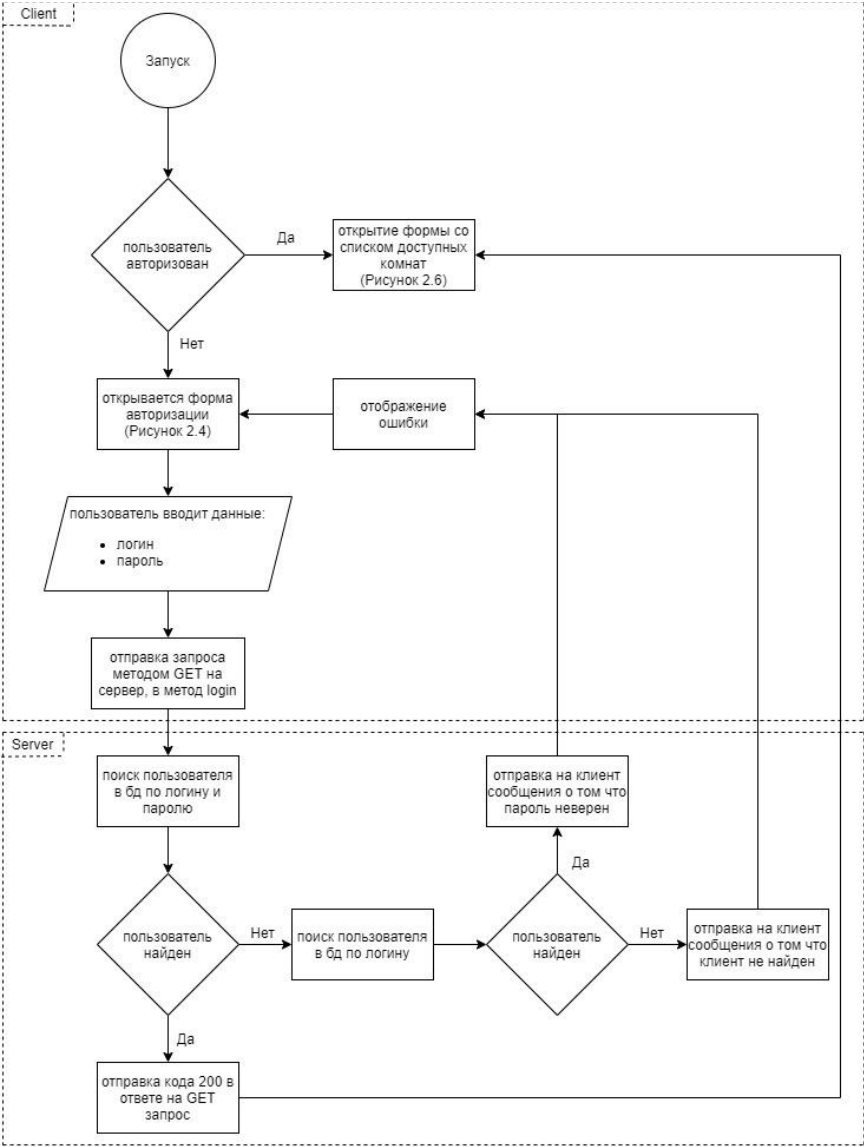


Рисунок 2.14 Алгоритм авторизации пользователя при загрузке приложения.

2.6.2 Алгоритм регистрации пользователя

Алгоритм инициируется по кнопке «Регистрация» на Форме авторизации (Рисунок 2.4).

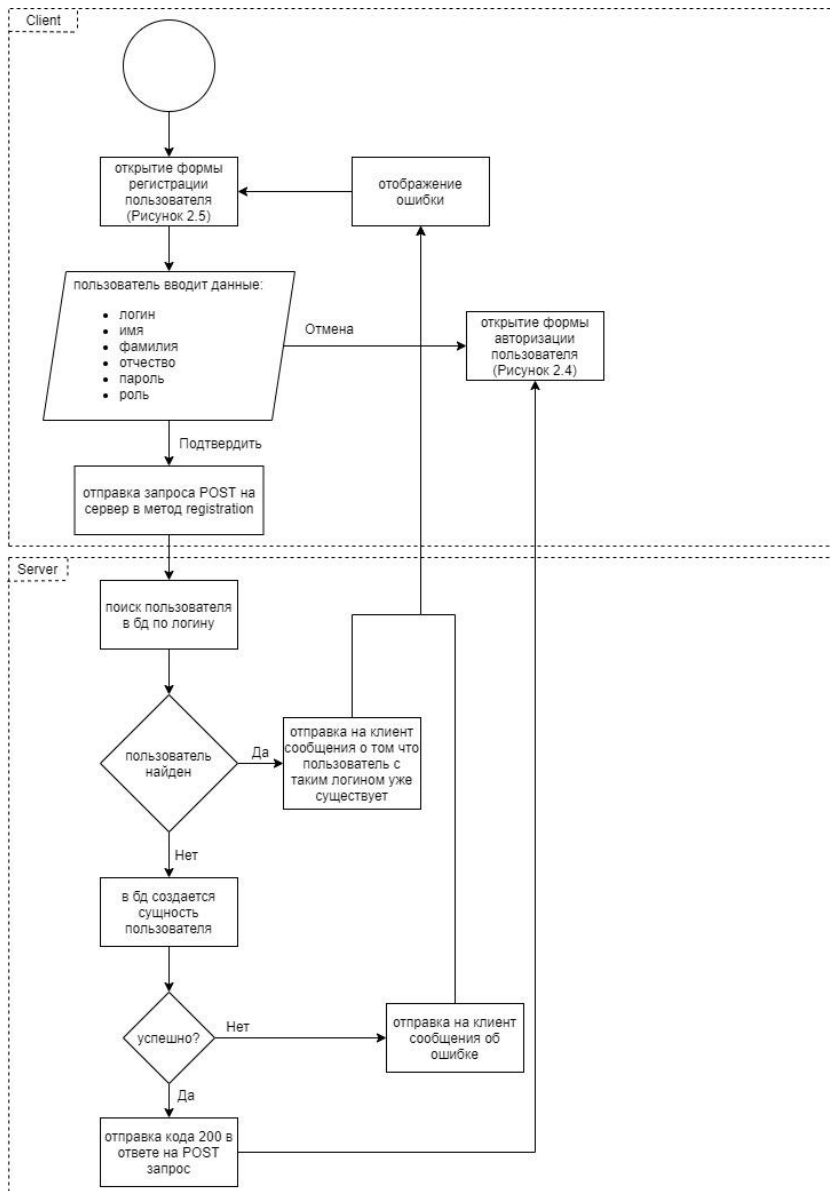


Рисунок 2.15 Алгоритм регистрации пользователя.

2.6.3 Алгоритм редактирования данных пользователя

Алгоритм инициируется нажатием на кнопку «Ред. дан. польз.» из выпадающего списка меню на форме просмотра доступных комнат (см. Рисунок 2.7).

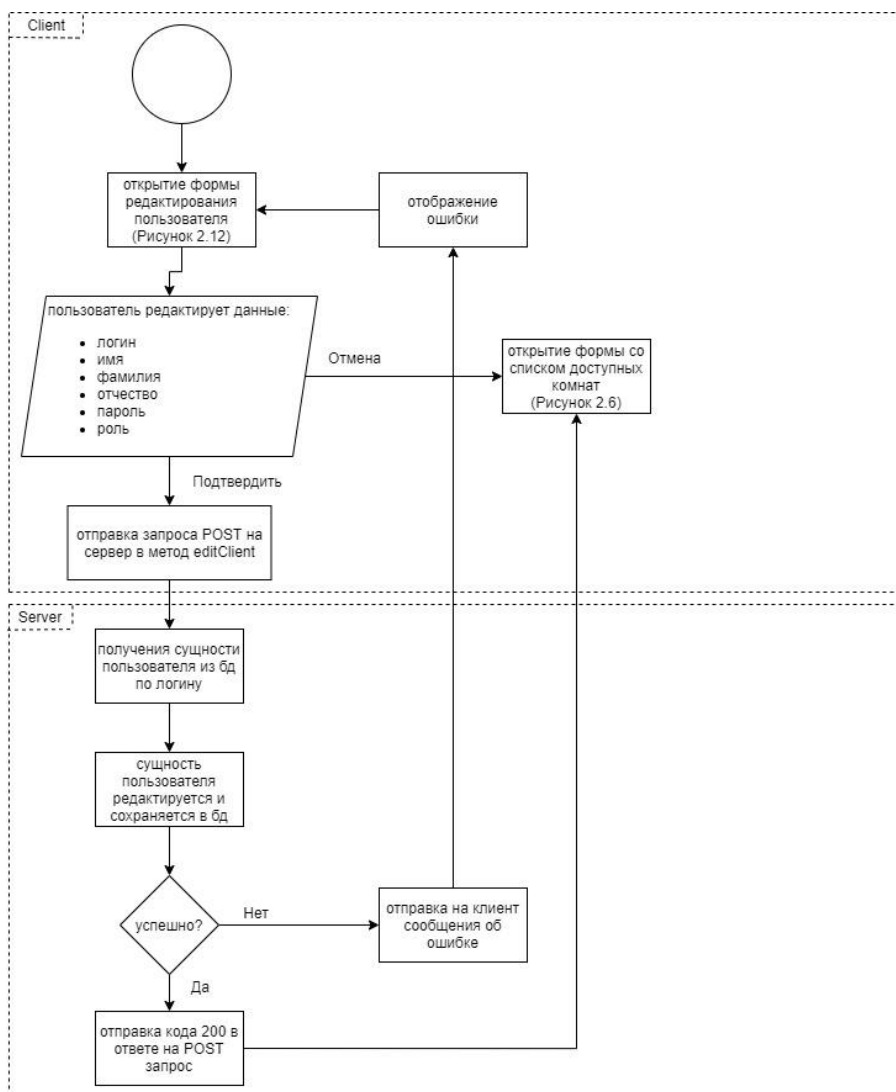


Рисунок 2.16 Алгоритм редактирования данных пользователя.

2.6.4 Алгоритм создания комнаты (группы)

Алгоритм инициируется нажатием на кнопку «Создать комнату» из выпадающего списка меню на форме просмотра доступных комнат (см. Рисунок 2.7). Доступно только пользователем с ролью «Ведущий».

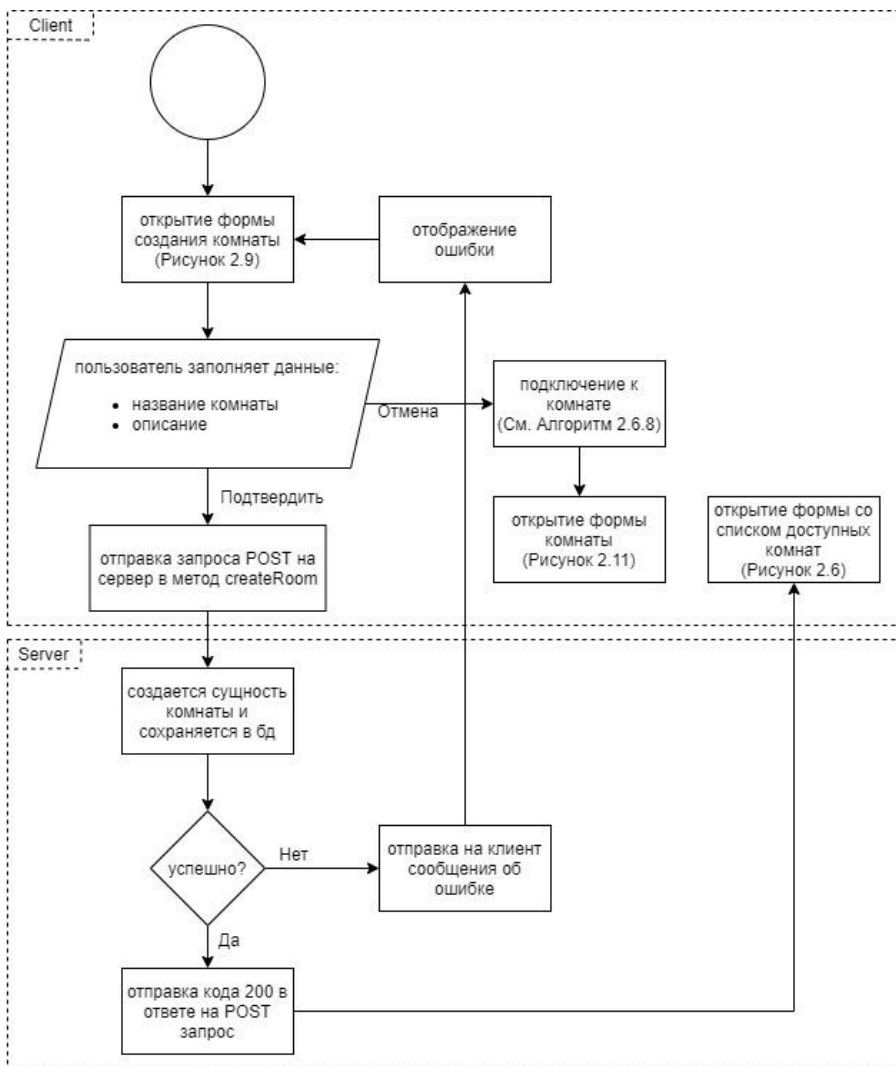


Рисунок 2.17 Алгоритм создания комнаты.

2.6.5 Алгоритм отображения списка доступных комнат (групп)

Алгоритм инициируется после успешной авторизации пользователя (См. Алгоритм 2.6.1).

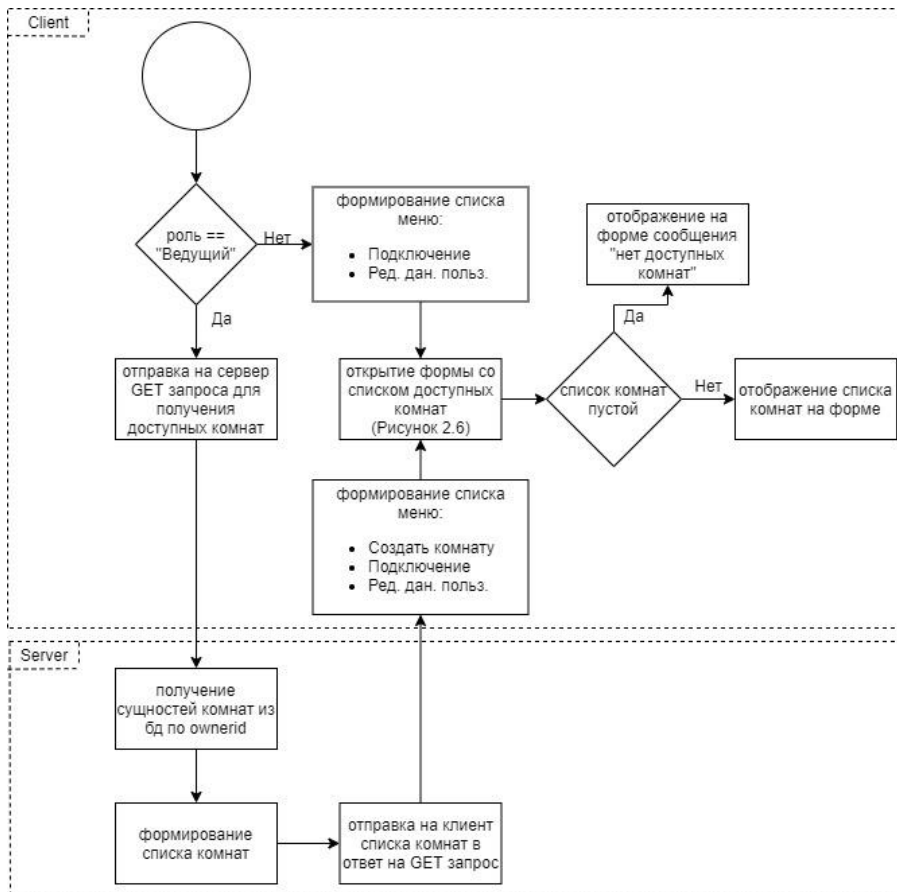


Рисунок 2.18 Алгоритм получения списка доступных комнат.

2.6.6 Алгоритм отображения информации о комнате (группе)

Алгоритм инициируется нажатием на элемент списка на форме доступных комнат (См. Рисунок 2.6).

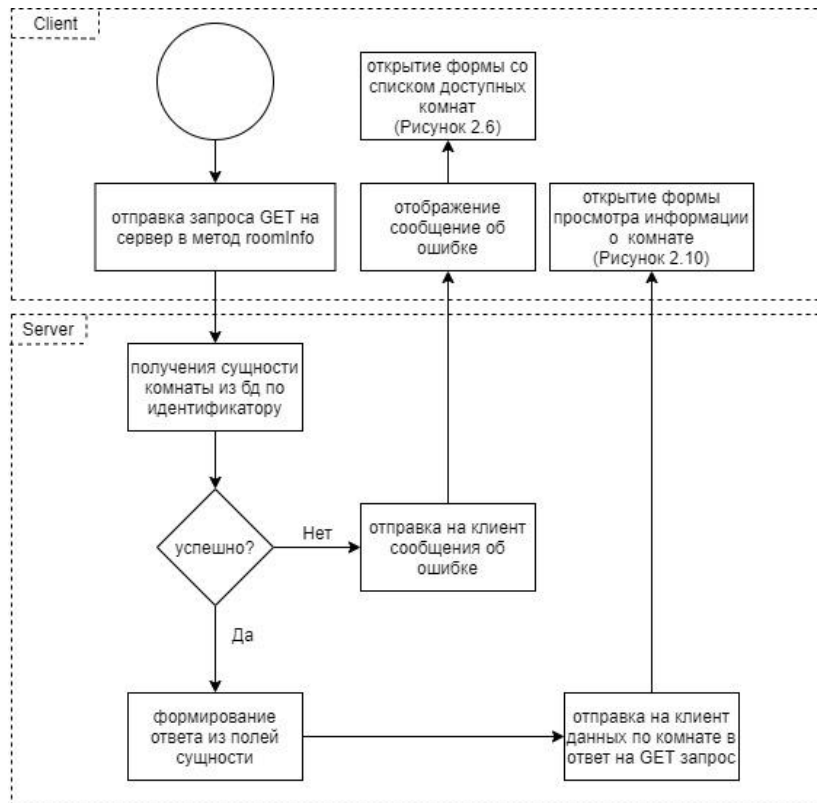


Рисунок 2.19 Алгоритм отображения информации о комнате.

2.6.7 Алгоритм редактирования комнаты (группы)

Алгоритм инициируется нажатием кнопки «Редактировать» на форме просмотра информации о комнате (См. Рисунок 2.10)

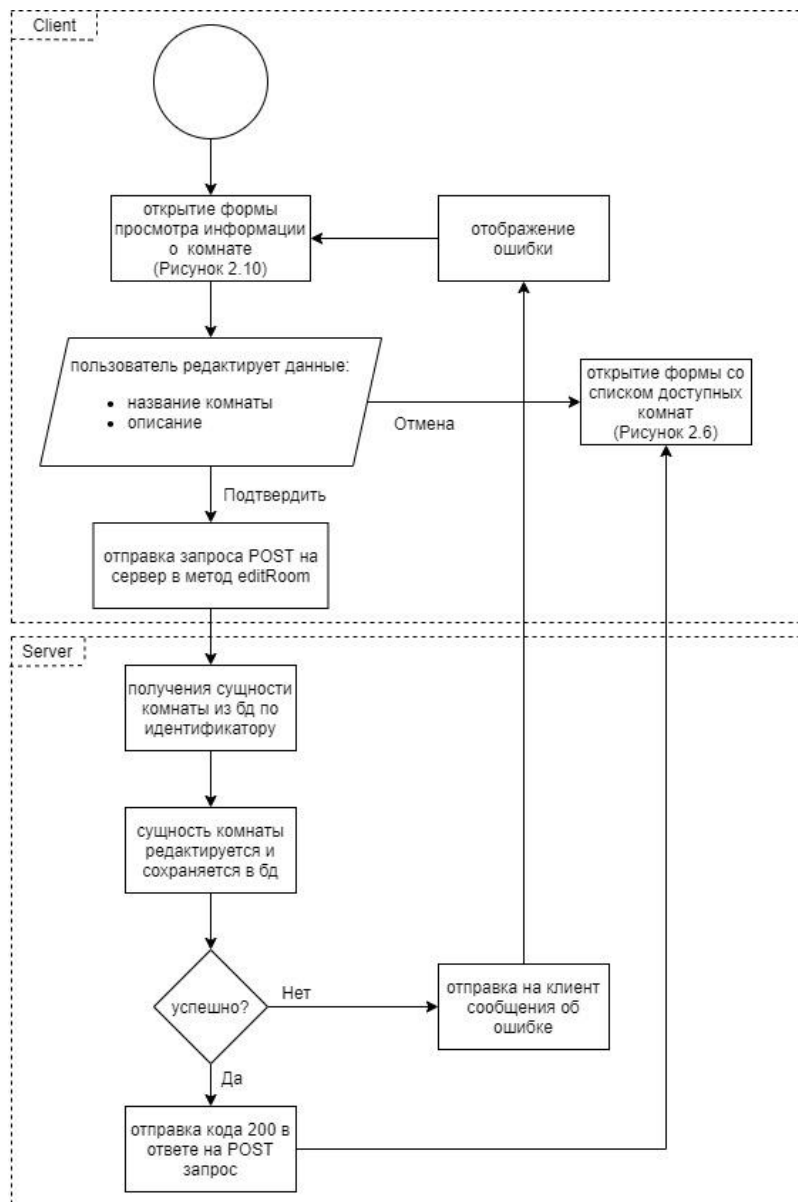


Рисунок 2.20 Алгоритм получения списка доступных комнат.

2.6.8 Алгоритм подключения к комнате (группе)

Алгоритм инициализируется по кнопке подключиться на форме подключения к комнате по идентификатору (См. Рисунок 2.8) или на форме просмотра информации о комнате (См. Рисунок 2.10), либо автоматически после успешного создания комнаты (См. Алгоритм 2.6.4).

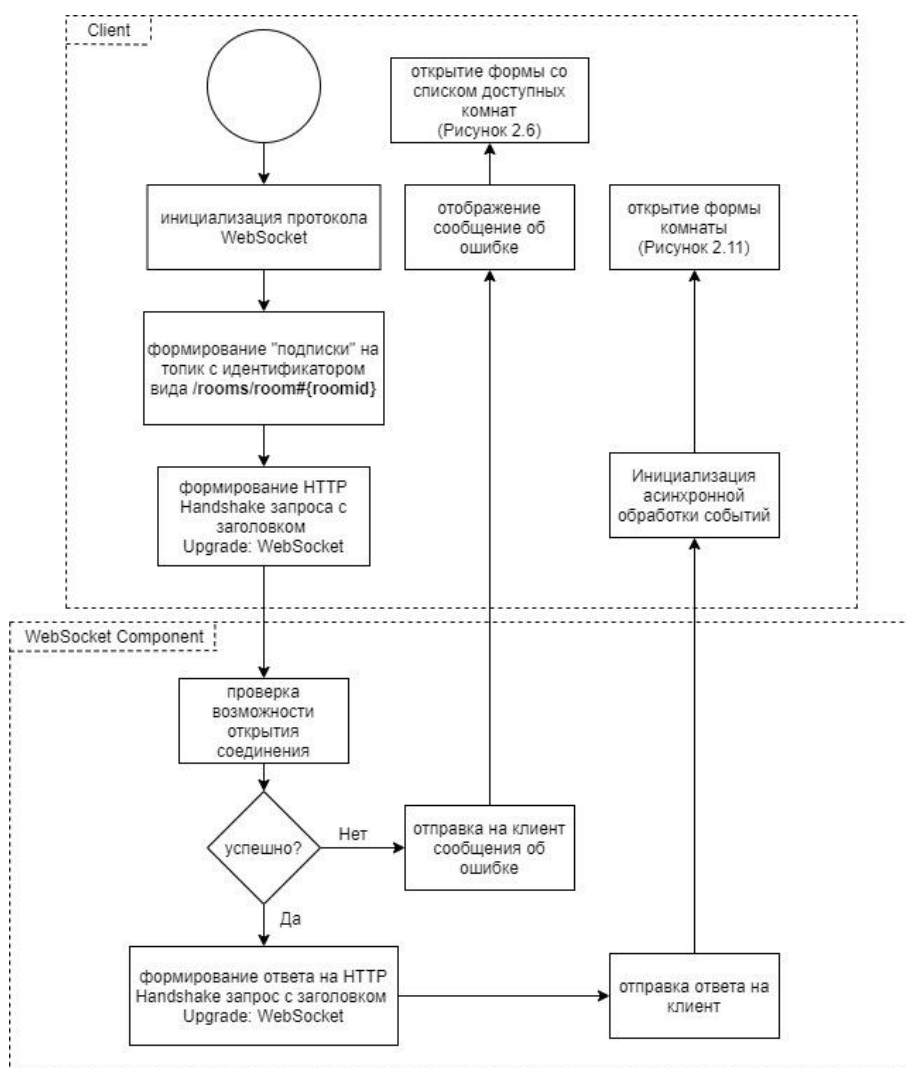


Рисунок 2.21 Алгоритм получения списка доступных комнат.

2.6.9 Алгоритм удаления комнаты (группы)

Алгоритм инициализируется долгим нажатием на элемент списка на форме доступных комнат (См. Рисунок 2.6).

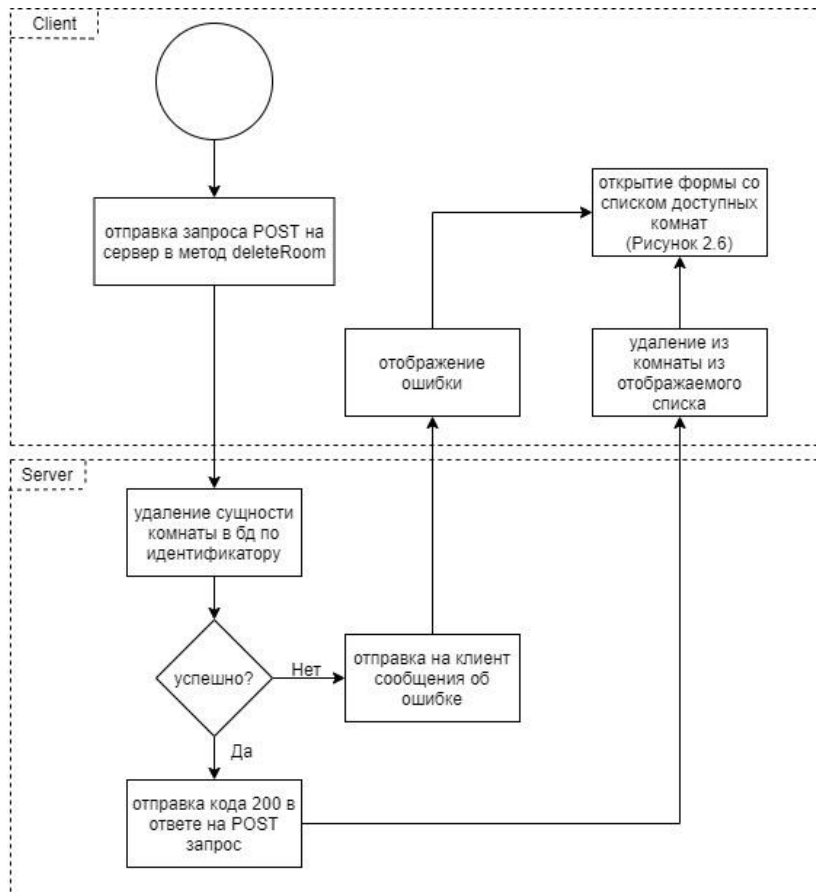


Рисунок 2.22 Алгоритм удаления комнаты.

2.6.10 Алгоритм отправки сообщений

Алгоритм инициализируется нажатием на кнопку отправить с символом «>» на форме комнаты (См. Рисунок 2.11).

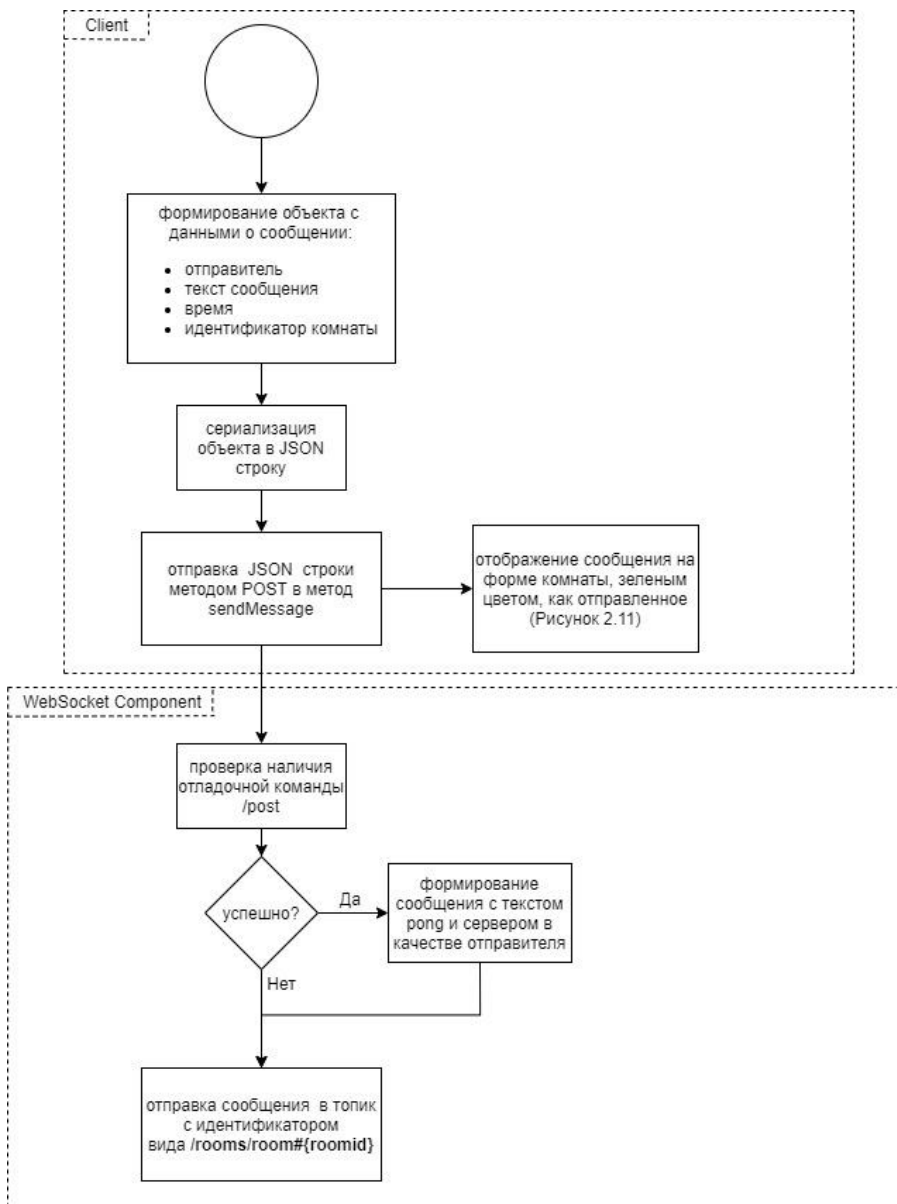


Рисунок 2.23 Алгоритм отправки сообщения.

2.6.11 Алгоритм обработки входящих сообщений

Алгоритм инициализируется при получении сообщения по протоколу WebSocket or WebSocket Component.

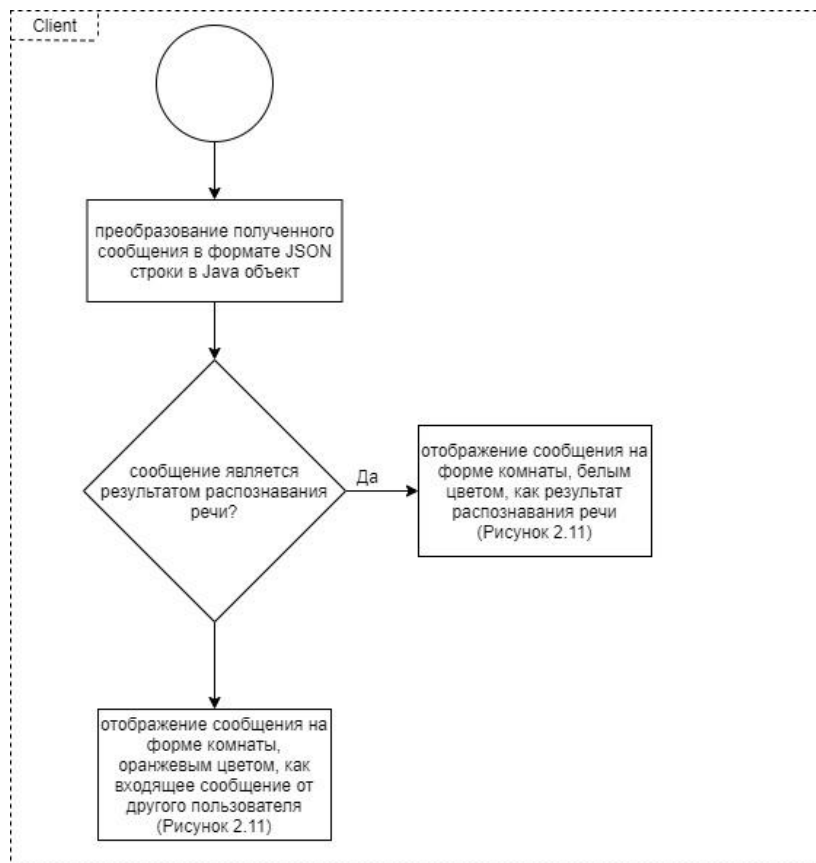


Рисунок 2.24 Алгоритм обработки входящего сообщения.

2.6.12 Алгоритм распознавания голоса в текст и отправка распознанного текста пользователем с ролью «Ведущий»

Алгоритм инициализируется по кнопке «Запустить расп.» на форме комнаты, алгоритм прерывается по кнопке «Остановить расп.» (См. Рисунок 2.13).

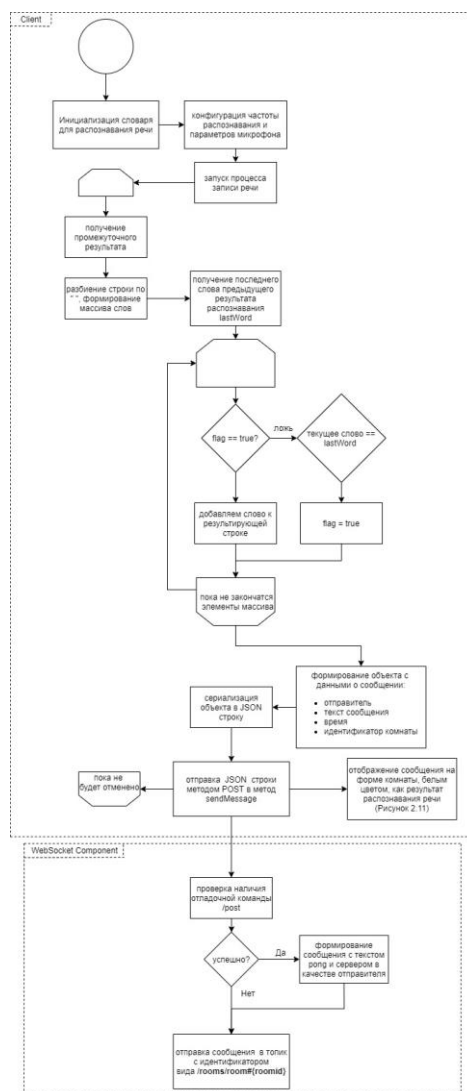


Рисунок 2.25 Алгоритм распознавания голоса в текст и отправки.

2.7 Разработка плана проведения тестирования и отладки.

Основная концепция тестирования заключается в последовательном выполнении выбранного алгоритма и сравнении фактического результата с ожидаемым.

Тестирование предназначено для выявления наличия ошибок с последующим их устранением. При сравнении фактического и ожидаемого результата проверяется выполнение алгоритма, а также точно полученных данных.

План проведения тестирования и отладки:

1. Выбор алгоритма из перечня реализованных алгоритмов.
2. Формирование ожидаемого результата
3. Последовательное выполнение алгоритма
4. Получение фактического результата.
5. Анализ полученного фактического результата.
 - а. Если полученный фактический результат не соответствует ожидаемому результату, проводится отладка алгоритма и исправление ошибок.
 - б. Выполняются пункты 2-5 пока фактический результат не будет соответствовать ожидаемому.
6. Пункты 1-6 циклически выполняются пока не будут пройдены все разработанные алгоритмы.

3 Экспериментальная часть

3.1 Реализация базы данных

Ниже представлена схема реализованной базы данных, а также краткое описание таблиц.

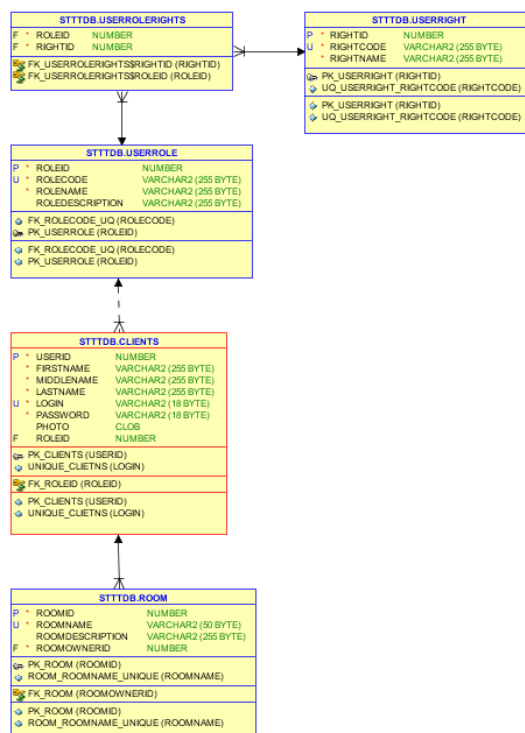


Рисунок 3.1 Схема реализованной базы данных.

CLIENTS – таблица содержащая зарегистрированных клиентов и основную информацию о них.

ROOM – таблица содержащая созданные комнаты клиентами комнаты, с информацией о имени комнаты и описанием, а так же ID пользователя создавшего комнату.

USERROLE – таблица содержащая роли пользователей. С информацией о имени роли и описанием.

USERRIGHT – таблица содержащая права пользователей. Содержит информацию о коде и имени права

USERROLERIGHTS – таблица являющаяся связующей для таблиц userrole и userright и служит для решения связи многие ко многим между двумя этими таблицами.

3.2 Реализация разработанных алгоритмов

Ниже представлено описание алгоритмов и необходимая последовательность действий.

3.2.1 Алгоритм авторизации пользователя при загрузке приложения

Алгоритм авторизации пользователя предназначен для авторизации пользователя по логину и паролю при запуске приложения.

Интерфейс:

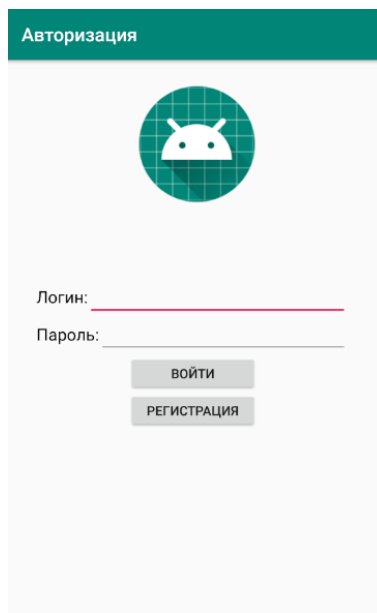


Рисунок 3.2 Окно авторизации пользователя

Основной сценарий:

1. При запуске приложения открывается окно авторизации пользователя (Рисунок 3.2).
2. Пользователь вводит авторизационные данные: логин и пароль. И нажимает на кнопку «войти».

3. На стороне клиентского приложения проходит валидация данных.
4. Авторизационные данные отправляются на сервер по http.

HTTP метод	URL
GET	http://host:port/SpeechToTextTranslator.server/login?login=login&password=password

5. На стороне сервера происходит проверка авторизационных данных с данными в базе данных
 - а. Если клиент с данным логином не найден – система возвращает код 200 и статус, обозначающий что клиент не найден. Переход на шаг 6.

Код ответа	Формат ответа
200	{ "status": "2", "client": {} }

- б. Если клиент с данным логином существует, но пароль не совпадает – система возвращает статус, обозначающий что пароль неверен. Переход на шаг 6.

Код ответа	Формат ответа
200	{ "status": "3", "client": {} }

- в. Если клиент с данным логином найден и пароль совпал – система получает всю информацию по пользователю и возвращает по протоколу http в приложение-клиент, вместе с статусом об успешной авторизации (по статусам авторизации см Таблица 1).

Код ответа	Формат ответа
200	{ "status": "1", "client": { "login": "login", "firstName": "firstName", "middleName": "middleName", "lastName": "lastName", "role": "role" } }

6. От сервера приходит ответ о результате авторизации.
- а. Если авторизация не была успешной, т. е. пришел статус 2 или 3 – клиент отображает соответствующее сообщение в зависимости от статуса авторизации (Таблица. 3.1)
 - б. Если авторизация была успешной, отображается представление со списком доступных комнат (Выполнение алгоритма 3.2.5).

Таблица 3.1 Таблица, содержащая статусы авторизации и тематические сообщения

Идентификатор статуса	Описание статуса	Сообщение
1	Авторизация успешна	-
2	Клиент не найден	Client not found
3	Неверный пароль	Incorrect password

3.2.2 Алгоритм регистрации пользователя

Алгоритм регистрации пользователя предназначен для первоначальной регистрации пользователя в системе.

Интерфейс:

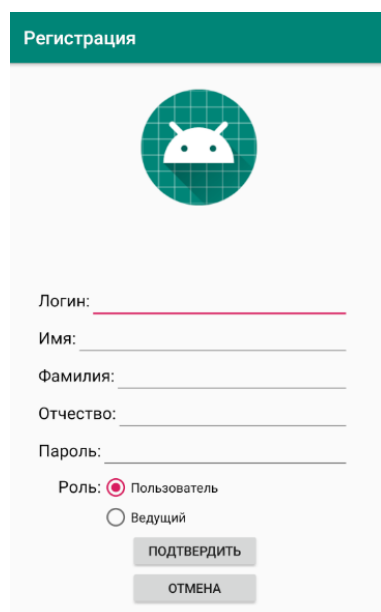


Рисунок 3.3 Окно регистрации пользователя

Основной сценарий:

1. Пользователь переходит в режим регистрации по нажатию кнопки «Регистрация» на форме окна авторизации (Рисунок 3.2).
2. Отображается окно регистрации (Рисунок 3.3).
3. Пользователь вводит регистрационные данные.
4. На стороне клиентского приложения проходит валидация данных.
5. Пользователь выбирает одно из доступных действий:
 - а. Если пользователь нажимает кнопку «подтвердить», переход на шаг 6.

- б. Если пользователь нажимает кнопку «отмена», возврат на форму авторизации (Рисунок 3.2), выполняется шаг 10.

6. Регистрационные данные отправляются на сервер по http.

HTTP метод	URL	Данные
POST	http://host:port/SpeechToTextTranslator.server/registration	{ "login": "login", "firstName": "firstName", "middleName": "middleName", "lastName": "lastName", "role": "role" }

7. На стороне сервера происходит проверка на наличие клиента с таким-же логином.

- а. Если клиент с таким логином уже существует, сервер возвращает код http 504 и сообщение о том, что клиент с таким логином уже существует. Переход на шаг 9.

Данные
{ "message": "Клиент с таким логином уже существует" }

8. На стороне сервера происходит сохранение клиента в базе данных.

- а. Если сохранение прошло успешно, переход на шаг 9.
- б. Если в процессе сохранения клиента возникли ошибки, система возвращает код http 504.

9. От сервера приходит ответ о результате регистрации.

- а. Если регистрация не была успешной (код http 504) – клиент отображает сообщение из тега "message". Если тег не найден или пустой, клиент отображает сообщение «network error».
- б. Если регистрация была успешной, отображается окно авторизации пользователя для дальнейшей авторизации (Рисунок 3.2)

10. Выполняется алгоритм «3.2.1 Алгоритм авторизации пользователя»

3.2.3 Алгоритм редактирования данных пользователя

Алгоритм редактирования данных пользователя предназначен для изменения данных пользователя, а также роли.

Интерфейс:

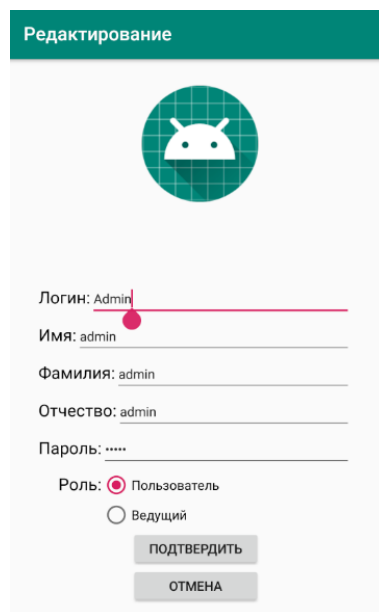
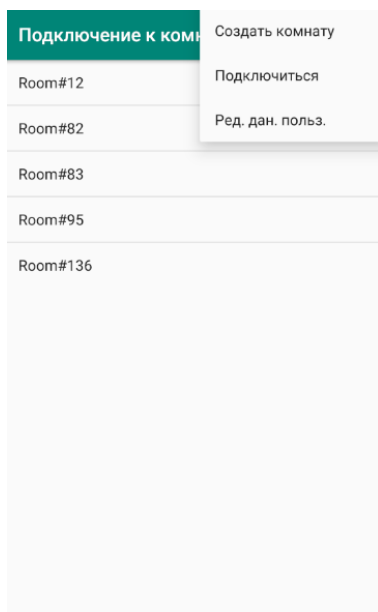


Рисунок 3.4 выпадающий список меню на форме доступных комнат.

Рисунок 3.5 Окно редактирования информации о клиенте.

Основной сценарий:

1. Пользователь нажимает на кнопку «Ред. Дан. Польз.» в выпадающем меню на форме доступных комнат (Рисунок 3.4).
2. Открывается форма редактирования данных о клиенте (Рисунок 3.5).
3. Пользователь редактирует данные.
4. Пользователь выбирает одно из доступных действий:
 - а. Если пользователь нажимает кнопку «подтвердить», переход на шаг 5.
 - б. Если пользователь нажимает кнопку «отмена», возврат на форму доступных комнат (Рисунок 3.4), выполняется шаг 9.

5. Регистрационные данные отправляются на сервер по http.

HTTP метод	URL	Данные
POST	http://host:port/SpeechToTextTranslator.server/editclient	{ "login": "login", "firstName": "firstName", "middleName": "middleName", "lastName": "lastName", "role": "role" }

6. На стороне сервера происходит сохранение клиента в базе данных.

- a. Если сохранение прошло успешно, переход на шаг 7.
- b. Если в процессе сохранения клиента возникли ошибки, система возвращает код http 504.

7. От сервера приходит ответ о результате редактирования клиента.

- a. Если редактирование клиента не было успешным (код http 504) – клиент отображает сообщение «network error».
- b. Если редактирование клиента было успешным, происходит возврат на форму доступных комнат (Рисунок 3.4).

8. Завершения сценария.

3.2.4 Алгоритм создания комнаты (группы)

Алгоритм предназначен для создания пользователями комнаты для дальнейшего подсоединения.

Интерфейс:

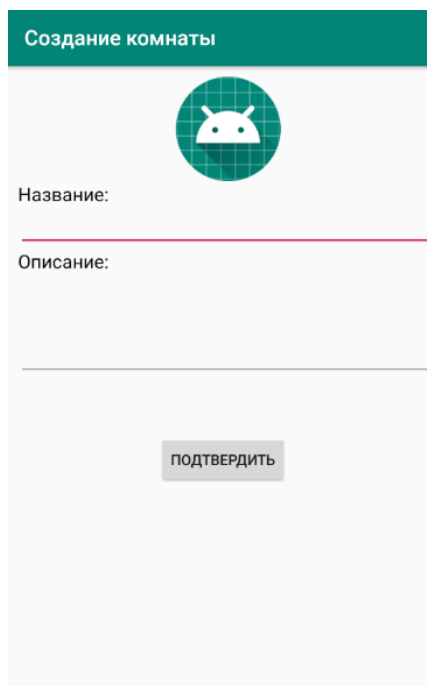


Рисунок 3.6 Окно создания комнаты

Основной сценарий:

1. Пользователь нажимает кнопку «Создание комнаты» в выпадающем меню на форме доступных комнат (Рисунок 3.4).
2. Открывается окно создания группы (Рисунок 3.6).
3. Пользователь заполняет данные: название и описание комнаты. Пользователь нажимает кнопку «Подтвердить»
4. Данные отправляются на сервер по http.

HTTP метод	URL	Данные
------------	-----	--------

POST	http://host:port/SpeechToTextTranslator.server/createRoom	{ "name": "name", "description": "description" }
------	---	---

5. На стороне сервера происходит проверка на наличие комнаты с таким-же названием.

- а. Если комната с таким названием уже существует, сервер возвращает код http 504 и сообщение о том, что комната с таким названием уже существует. Переход на шаг 7.

Данные
{ "message": "Комната с таким названием уже существует" }

6. На стороне сервера происходит сохранение комнаты.

- а. Если сохранение прошло успешно, переход на шаг 6.
- б. Если при сохранении комнаты возникли ошибки, система возвращает код http 504, переход на шаг

7. От сервера приходит ответ о результате редактирования клиента.

- а. Если редактирование клиента не было успешным (код http 504) – клиент отображает сообщение из тега "message". Если тег не найден или пустой, клиент отображает сообщение «network error».
- а. Если редактирование клиента было успешным, происходит подключение к созданной комнате, выполнение алгоритма 3.2.8.

8. Завершение сценария.

3.2.5 Алгоритм отображения списка доступных комнат (групп)

Алгоритм предназначен для отображения списка доступных для пользователя комнат. На данном этапе в этот список попадают комнаты, для которых создателем является текущий клиент.

Интерфейс:

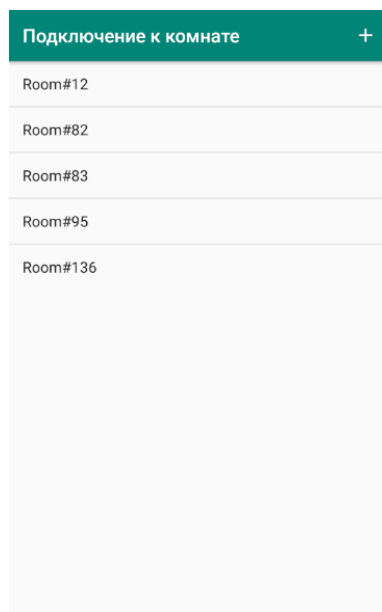


Рисунок 3.7 Окно со списком доступных комнат.

Основной сценарий:

1. Пользователь успешно завершает авторизацию на форме авторизации пользователя (Рисунок 3.2).
2. На сервер отправляется http запрос.

HTTP метод	URL
GET	http://host:port/SpeechToTextTranslator.server/getallrooms?login=login

3. На сервере отбираются комнаты, у которых создателем является клиент с логином, совпадающим с логином из запроса.

- а. Если комнаты не найдены, система формирует пустой список.

4. Сервер возвращает код http 200 и список доступных комнат

HTTP код	Данные при наличии комнат	Данные при отсутствии комнат
200	<pre>{ "rooms": [{ "roomid": "roomid", "roomname": "roomname", "roomdescription": "roomdescription" }, { "roomid": "roomid", "roomname": "roomname", "roomdescription": "roomdescription" }, ...] }</pre>	<pre>{ "rooms": [] }</pre>

5. Клиент анализирует ответ от сервера.

- а. Если в ответе вернулся пустой список доступных комнат, система отображает сообщение «Не найдено доступных комнат» на форме доступных комнат (Рисунок 3.7).
- б. Если в ответе от сервера вернулся не пустой список доступных комнат, сервер отображает доступные комнаты на форме доступных комнат (Рисунок 3.7)

6. Клиент формирует выпадающее меню (Рисунок 3.5)

- а. Если пользователь обладает ролью «Ведущий», формируется меню, содержащее пункты: «Создать комнату», «Подключиться к комнате», «Ред. Дан. Польз.»

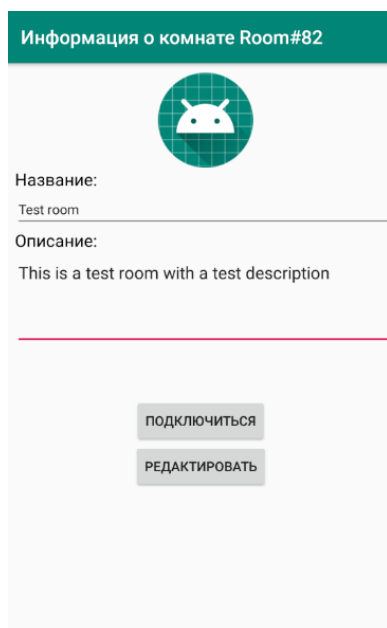
- б. Если пользователь обладает ролью «Пользователь», формируется меню, содержащее пункты «Подключиться к комнате», «Ред. Дан. Польз.»

7. Завершение сценария.


3.2.6 Алгоритм отображения информации о комнате (группе)

Алгоритм предназначен для просмотра информации о комнате при выборе из списка доступных комнат, и редактирования комнаты или подключения к ней.

Интерфейс:



Информация о комнате Room#82



Название:
Test room

Описание:
This is a test room with a test description

ПОДКЛЮЧИТЬСЯ

РЕДАКТИРОВАТЬ

Рисунок 3.8 Окно просмотра информации о комнате.

Основной сценарий:

1. Пользователь выбирает комнату из списка доступных комнат с помощью нажатия на элемент списка на форме доступных комнат (Рисунок 3.7).

2. На сервер отправляется http запрос для получения полной информации по комнате.

HTTP метод	URL
GET	http://host:port/SpeechToTextTranslator.server/getroom?roomid=roomid

3. На стороне сервера происходит поиск комнаты по идентификатору комнаты.

- Если комната не найдена, или в процессе поиска комнаты возникли ошибки. Сервер возвращает код http 504, переход на шаг 4.
- Если комната найдена, сервер формирует ответ и посылает на клиент код http 200 вместе с информацией о комнате.

Данные
{ "roomid": "roomid", "roomname": "roomname", "roomdescription": "roomdescription" }

4. Клиент анализирует ответ от сервера, полученный на шаге 3.
- Если в процессе получения информации о комнате произошла ошибка (код http 504), система отображает ошибку «network error». Возврат на форму доступных комнат (Рисунок 3.7). Переход на шаг 7.
5. Отображение формы просмотра информации о комнате (Рисунок 3.8).
6. Пользователь выбирает одно из доступных действий:
- Если пользователь нажимает кнопку «подключиться», тогда выполняется алгоритм 3.2.8.

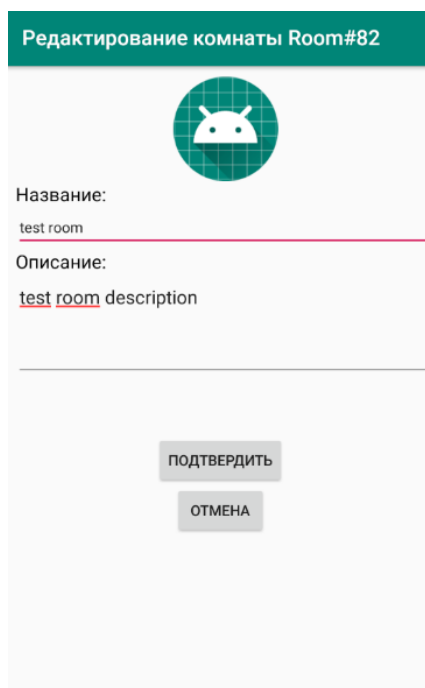
- б. Если пользователь нажимает кнопку «редактировать», тогда выполняется алгоритм 3.2.7.

7. Завершение сценария


3.2.7 Алгоритм редактирования комнаты (группы)

Алгоритм предназначен для изменения пользователями данных о комнате.

Интерфейс:



Редактирование комнаты Room#82



Название:
test room

Описание:
test room description

ПОДТВЕРДИТЬ

ОТМЕНА

Рисунок 3.9 Окно редактирования комнаты

Основной сценарий:

1. Пользователь нажимает кнопку «Редактировать» на форме просмотра информации о комнате (Рисунок 3.8).
2. Открывается форма редактирования комнаты (Рисунок 3.9).
3. Пользователь редактирует данные комнаты.

4. Пользователь выбирает одно из доступных действий:
 - а. Если пользователь нажимает кнопку «подтвердить», переход на шаг 5.
 - б. Если пользователь нажимает кнопку «отмена», переход на шаг 8.

5. Данные отправляются на сервер по http.

HTTP метод	URL	Данные
POST	http://host:port/SpeechToTextTranslator.server/editroom	{ "name": "name", "description": "description" }

6. На сервере происходит обновление данных комнаты.

- а. Если сохранение было успешным, сервер возвращает код http 200 и обновленные данные о комнате

Данные
{ "roomid": "roomid", "roomname": "roomname", "roomdescription": "roomdescription" }

- б. Если в процессе сохранения комнаты произошла ошибка, сервер возвращает код http 504.

7. От сервера приходит ответ о результате редактирование комнаты.

- а. Если редактирование не было успешным (код http 504) – клиент отображает сообщение «network error».

8. Открытие формы просмотра информации о комнате (Рисунок 3.8)

- а. Если на шаге 7, был получен код http 200, обновление данных на форме, данными, полученными на шаге 6.а.

9. Завершение сценария.

3.2.8 Алгоритм подключения к комнате (группе)

Алгоритм, предназначенный для подключения пользователей к комнате, для обеспечения взаимодействия посредством обмена сообщениями для клиентов, подключенных к комнате.

Интерфейс:

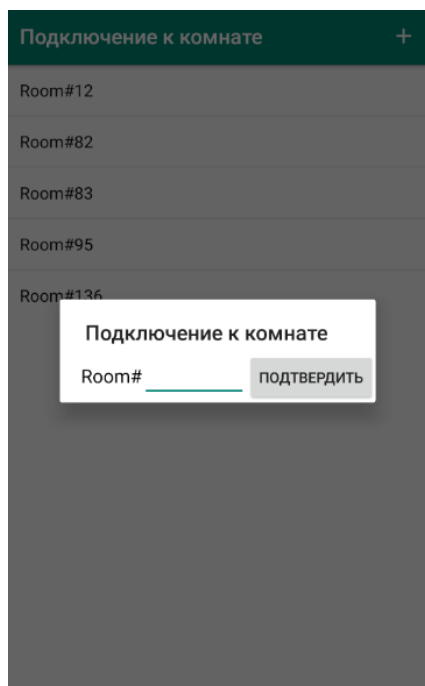


Рисунок 3.10 Подключение к комнате по идентификатору.

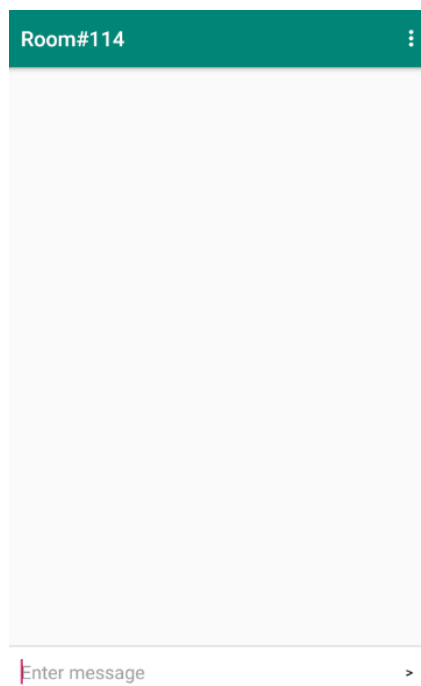


Рисунок 3.11 Окно комнаты.

Основной сценарий:

1. Для инициализации алгоритма пользователь совершает одно из доступных действий:
 - а. Если пользователь нажимает на кнопку «Подключиться» в выпадающем меню на форме доступных комнат (Рисунок 3.4.), то открывается модальная форма подключения к комнате (Рисунок 3.10).

- b. Пользователь вводит идентификатор комнаты и нажимает кнопку «Подтвердить», переход на шаг 2
 - c. Если пользователь нажимает кнопку «Подключиться» на форме просмотра информации о комнате (Рисунок 3.8), переход на шаг 2.
 - d. Если ранее произошло успешное создание комнаты в алгоритме 3.2.4.
- 2. Формируется подписка на очередь с идентификатором вида /rooms/room#{roomid}.
- 3. На сервер в WebSocket Component отправляется запрос на формирование подписки по адресу ws://host:port/ws/websocket
- 4. Клиент анализирует ответ от WebSocket Component:
 - a. Если в процессе формирования подписки возникли ошибки, клиент отображает ошибку «network error», переход на шаг 6.
- 5. Открывается окно комнаты (Рисунок 3.11).
- 6. Запуск асинхронного процесса обработки сообщений. Если из очереди приходит сообщение, выполнение сценария 3.2.11.
- 7. Завершение сценария.

3.2.9 Алгоритм удаления комнаты (группы)

Алгоритм, предназначенный для удаления комнат пользователем, который является создателем комнаты.

Интерфейс:

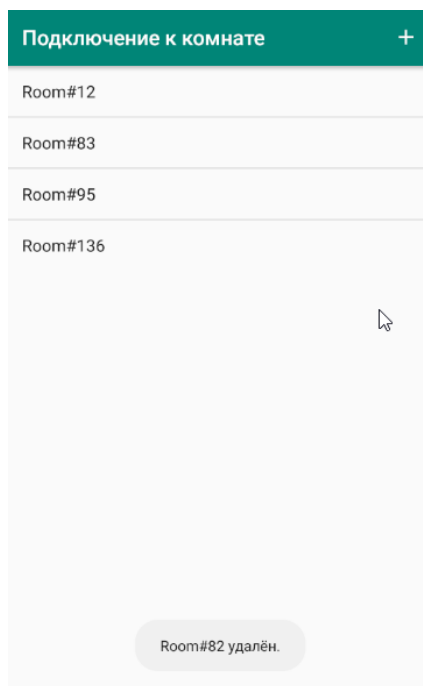


Рисунок 3.12 Сообщение об удалении комнаты на форме доступных комнат

Основной сценарий:

1. Пользователь совершает длинное нажатие на элемент списка на форме доступных комнат (Рисунок 3.7).
2. На сервер отправляется http запрос

HTTP метод	URL	Данные
POST	http://host:port/SpeechToTextTranslator.server/deleteroom	{ "roomid": "roomid" }

3. На стороне сервера происходит удаление записи о комнате из базы данных.

- a. Если во время удаления комнаты произошла ошибка, система возвращает код http 504.
 - b. Если во время удаления комнаты не возникло ошибок, система возвращает код http 200.
- 4. Клиент анализирует ответ от сервера.
 - a. Если при удалении возникли ошибки (код http 504) – клиент отображает сообщение «network error», переход на шаг 7.
 - b. Если удаление комнаты было успешным, переход на шаг 5.
- 5. Удаление выбранного элемента из списка на форме доступных комнат (Рисунок 3.7).
- 6. Отображение сообщения «Room#{roomid} удален».
- 7. Завершение сценария.

3.2.10 Алгоритм отправки сообщений

Алгоритм предназначен для отправки сообщений пользователя, на устройства пользователей, подключенных к группе пользователя, отправившего сообщение.

Интерфейс:

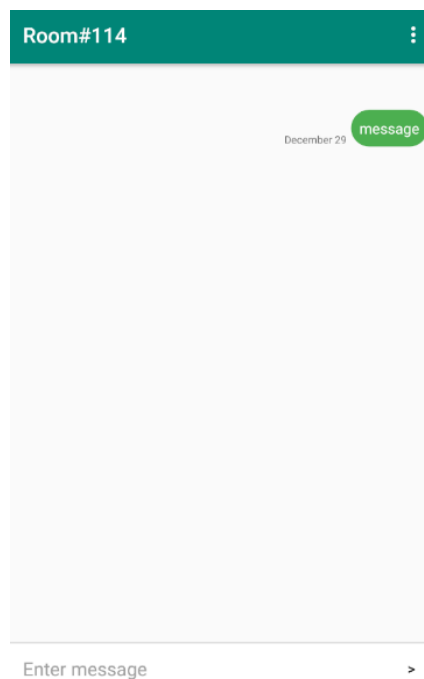


Рисунок 3.13 Окно комнаты с отображение отправленного сообщения.

Основной сценарий:

1. Пользователь вводит текст сообщения и нажимает на кнопку отправки сообщения с символом «>».
2. На WebSocket Component отправляется ws запрос, содержащий информацию о сообщении.

Протокол	URL	Данные
WS	ws://host:port/ws/websocket/topic/sendMessage	<pre>{ "message": "message", "sender": { "login": "login", "firstName": "firstName", "middleName": "middleName", "lastName": "lastName" }, "createdAt": "createdAt", "channelId": "channelId", "isRecognized": "isRecognized" }</pre>

3. На стороне WebSocket Component'a производится анализ сообщения
 - а. Если сообщение содержит текст отладочной «/ping», тогда дополнительно формируется сообщение с текстом «pong», в качестве отправителя указывается «Server», переход на шаг 4.
4. Сообщение помещается в очередь с идентификатором /rooms/room#{roomid}.
5. Сообщение из очереди отправляется на все устройства, которые подписаны на очередь с идентификатором /rooms/room#{roomid}.
6. Клиент анализирует отправленное сообщение.
 - а. Если сообщение содержит тег isRecognized = true, сообщение отображается на форме окна комнаты черным текстом с белым фоном по центру экрана (Рисунок 3.13).
 - б. Если сообщение не содержит тег isRecognized = true отображается на форме окна комнаты зеленым цветом (Рисунок 3.13).
7. Завершение сценария.

3.2.11 Алгоритм обработки входящих сообщений

Алгоритм предназначен для обработки поступающих из очереди сообщений и отображения их на форме окна комнаты (Рисунок 3.13).

Интерфейс:

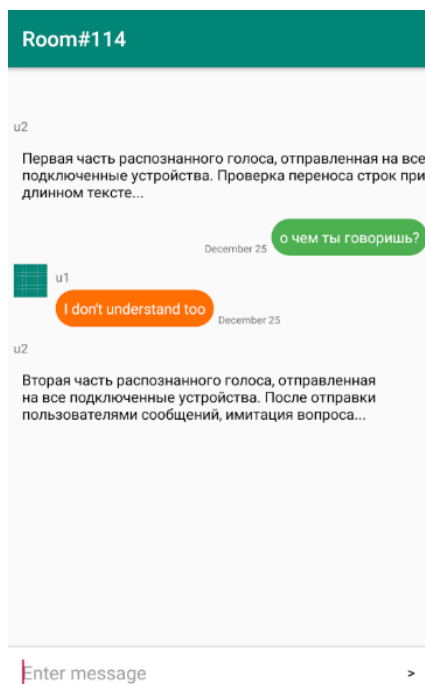


Рисунок 3.14 Окно комнаты с отображением входящих сообщений.

Основной сценарий:

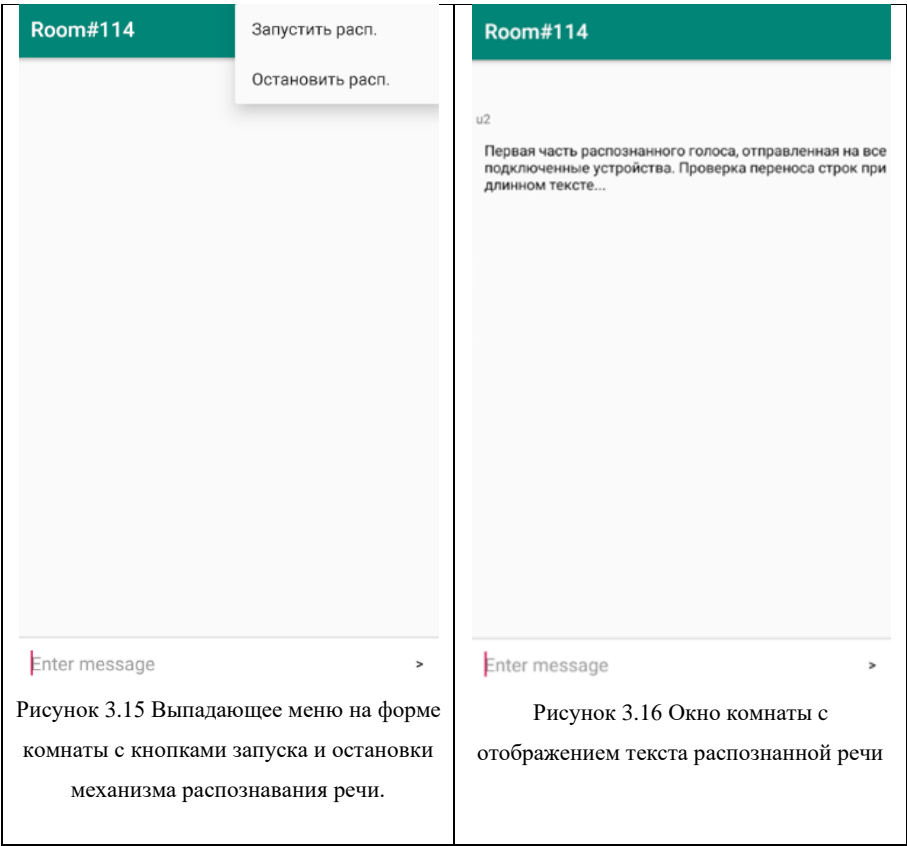
1. На клиент приходит сообщение из очереди, на которую была оформлена подписка в алгоритме 3.2.8.
2. На стороне клиента анализируется входящее сообщение.
 - а. Если логин отправителя совпадает с логином текущего пользователя, переход на шаг 4.
 - б. Если в входящем сообщении `tag isRecognized = true`, переход на шаг 3.

- с. Если пункты 2.а и 2.б не выполняются, сообщение добавляется с писк полученных сообщений и отображается на форме окна комнаты оранжевым цветом в левой части экрана, как входящее сообщение (Рисунок 3.14), далее переход на шаг 4.
3. Происходит поиск последнего сообщения, полученного в рамках текущей сессии.
- а. Если последнее сообщение содержит тег `isRecognized = true`, текст сообщения дополняется текстом полученного сообщения.
 - б. Если последнее сообщение не содержит тег `isRecognized = true`, полученное сообщение добавляется в конец списка полученных сообщений и отображается на форме окна формы текстом черного цвета с белым фоном и выравниванием по ширине (Рисунок 3,14).
4. Завершение сценария.

3.2.12 Алгоритм распознавания голоса в текст и отправка распознанного текста пользователем с ролью «Ведущий»

Алгоритм предназначен для распознавания голоса ведущего в текст и рассылки на устройства, подключенные к комнате ведущего.

Интерфейс:



Основной сценарий:

1. Пользователь нажимает кнопку «Запустить расп.» в выпадающем меню на форме комнаты (Рисунок 3.15).
2. Инициализируется словарь языка для распознавания.
3. Конфигурируются параметры механизма распознавания голоса.

4. Механизм распознавания речи запускается
5. При фиксации голоса библиотека Android Speech, производит предварительное распознавание полученного фрагмента звуковой дорожки и возвращает объект содержащий промежуточный результат.
6. Система анализирует полученный объект
 - a. Если в объекте, содержащем промежуточный результат, содержится результат распознавания с ключом «results_recognition», система считает этот результат конечным результатом распознавания.
 - b. Иначе переход на шаг 9.
7. Система анализирует предыдущий результат распознавания.
 - a. Если предыдущего результата нет, система формирует сообщение, где результат распознавания указывается в качестве текста сообщений, переход на шаг 8.
 - b. Если предыдущий результат распознавания существует и отличен от пустой строки, система получает последнее слово из строки предыдущего результата распознавания.
 - c. Анализируется текущий результат распознавания.
 - d. Ищется вхождение последнего слова предыдущего результата в текущем результате.
 - e. Все символы до позиции вхождения последнего слова в строке текущего результата распознавания, удаляются.
 - f. Система формирует сообщение, где текущий, измененный результат распознавания указывается в качестве текста сообщения, переход на шаг 8
8. Клиент сообщение. Выполняется алгоритм 3.2.10.
9. Переход на шаг 5.

Дополнительный сценарий:

1. Пользователь нажимает кнопку «Остановить расп.» в выпадающем меню на форме комнаты (Рисунок 3.15).
2. Выполнение основного сценария прерывается.
3. Механизм распознавания речи останавливается

3.3 Тестирование и отладка

Ниже приведен перечень алгоритмов и результат их выполнения.

3.3.1 Тестирование алгоритма авторизации пользователя

Алгоритм:

Алгоритм выполнения описан в пункте «3.2.1 Алгоритм авторизации пользователя»

Ожидаемый результат:

Авторизация пользователя проходит успешно по с помощью авторизационных данных

Фактический результат:

Фактический результат, полученный после выполнения алгоритма, соответствует ожидаемому результату.

3.3.2 Тестирование алгоритма регистрации пользователя

Алгоритм:

Алгоритм выполнения описан в пункте «3.2.2 Алгоритм регистрации пользователя»

Ожидаемый результат:

Пользователь зарегистрирован, в базе данных появилась запись с данными пользователя. При регистрации пользователя с таким же логином выдается ошибка о существовании пользователя с таким логином.

Фактический результат:

Фактический результат, полученный после выполнения алгоритма, соответствует ожидаемому результату.

3.3.3 Тестирование алгоритма редактирования данных пользователя

Алгоритм:

Алгоритм выполнения описан в пункте «3.2.3 Алгоритм редактирования данных пользователя»

Ожидаемый результат:

Данные пользователя в базе данных меняются на указанные.

Фактический результат:

Фактический результат, полученный после выполнения алгоритма, соответствует ожидаемому результату.

3.3.4 Тестирование алгоритма создания комнаты (группы)

Алгоритм:

Алгоритм выполнения описан в пункте «3.2.4 Алгоритм создания комнаты (группы)»

Ожидаемый результат:

В базе данных создается запись, содержащая данные о комнате, включая имя комнаты и описание. При попытке повторно создать комнату с таким же именем пользователю должно выдаваться сообщения о существовании комнаты с таким именем.

Фактический результат:

Фактический результат, полученный после выполнения алгоритма, соответствует ожидаемому результату.

3.3.5 Тестирование алгоритма отображения списка доступных комнат (групп)

Алгоритм:

Алгоритм выполнения описан в пункте «3.2.5 Алгоритм отображения списка доступных комнат (групп)»

Ожидаемый результат:

Форма со списком доступных комнат корректно отображается (Рисунок 3.7) и содержит список комнат создателем которых является текущий пользователь.

Фактический результат:

Фактический результат, полученный после выполнения алгоритма, соответствует ожидаемому результату.

3.3.6 Тестирование алгоритма отображения информации о комнате (группе)

Алгоритм:

Алгоритм выполнения описан в пункте «3.2.6 Алгоритм отображения информации о комнате (группе)»

Ожидаемый результат:

Форма отображения информации о комнате корректно отображается и содержит информацию о комнате, содержащуюся в базе данных

Фактический результат:

Фактический результат, полученный после выполнения алгоритма, соответствует ожидаемому результату.

3.3.7 Тестирование алгоритма редактирования комнаты (группы)

Алгоритм:

Алгоритм выполнения описан в пункте «3.2.7 Алгоритм редактирования комнаты (группы)»

Ожидаемый результат:

После выполнения алгоритма данные о комнате в базе данных корректно обновляются. На форме просмотра информации о комнате данные так-же обновляются

Фактический результат:

Фактический результат, полученный после выполнения алгоритма, соответствует ожидаемому результату.

3.3.8 Тестирование алгоритма подключения к комнате (группе)

Алгоритм:

Алгоритм выполнения описан в пункте «3.2.8Алгоритм подключения к комнате (группе)»

Ожидаемый результат:

После выполнения алгоритма корректно открывается форма окна комнаты (Рисунок 3.11). Корректно создается подписка на очередь с идентификатором /rooms/room#{roomid}

Фактический результат:

Фактический результат, полученный после выполнения алгоритма, соответствует ожидаемому результату.

3.3.9 Тестирование алгоритма удаления комнаты (группы)

Алгоритм:

Алгоритм выполнения описан в пункте «3.2.9 Алгоритм удаления комнаты (группы)»

Ожидаемый результат:

После выполнения алгоритма из базы данных удаляется запись о выбранной комнате. На форме просмотра доступных комнат (Рисунок 3.7) отображается корректный список, с учетом удаленной комнаты.

Фактический результат:

Фактический результат, полученный после выполнения алгоритма, соответствует ожидаемому результату.

3.3.10 Тестирование алгоритма отправки сообщений

Алгоритм:

Алгоритм выполнения описан в пункте «3.2.10 Алгоритм отправки сообщений»

Ожидаемый результат:

При отправке сообщений все пользователи, подключенные к группе текущего пользователя, корректно получают сообщения. Отправленные сообщения корректно отображаются в окне комнаты.

Фактический результат:

Фактический результат, полученный после выполнения алгоритма, соответствует ожидаемому результату.

3.3.11 Тестирование алгоритма обработки входящих сообщений

Алгоритм:

Алгоритм выполнения описан в пункте «3.2.11 Алгоритм обработки входящих сообщений»

Ожидаемый результат:

Сообщения, отправленные в очередь, на которую подписан текущий пользователь корректно доставляются, и корректно отображаются на форме комнаты.

Фактический результат:

Фактический результат, полученный после выполнения алгоритма, соответствует ожидаемому результату.

3.3.12 Тестирование алгоритма распознавания голоса в текст и отправка распознанного текста пользователем с ролью «Ведущий»

Алгоритм:

Алгоритм выполнения описан в пункте «3.2.12 Алгоритм распознавания голоса в текст и отправка распознанного текста пользователем с ролью «Ведущий»»

Ожидаемый результат:

Запуск и остановка механизма распознавания голоса происходит корректно. Распознанный текст корректно доставляется на устройства пользователей подключенных к той-же комнате. Точность распознаваемой речи больше 80%.

Фактический результат:

Фактический результат, полученный после выполнения алгоритма, соответствует ожидаемому результату. Точно распознаваемого текста не ниже ожидаемых 80%

Заключение

Список использованной литературы

1. Yandex SpeechKit Документация – [Электронный ресурс]:
<https://cloud.yandex.ru/docs/speechkit/?redirected=true> (Дата обращения: 16.11.2020);
2. Google cloud speech api documentation – [Электронный ресурс]:
<https://cloud.google.com/text-to-speech/docs?hl=ru> (Дата обращения: 16.11.2020);
3. Обработка речи в Android Xamarin – [Электронный ресурс]:
<https://docs.microsoft.com/ru-ru/xamarin/android/platform/speech> (Дата обращения: 16.11.2020);
4. Интервью с Акмалом Артиковым о создании приложения Яндекс.Разговор – [Электронный ресурс]:
<https://miptstream.ru/2015/09/03/deaf-interview/> (Дата обращения: 16.11.2020);
5. Consultez nos pages d'aide | Rogervoice Help Center – [Электронный ресурс]: <https://help.rogervoice.com/en/collections/99303-consultez-nos-pages-d-aide> (Дата обращения: 16.11.2020);
6. LanguageTool – [Электронный ресурс]:
<https://ru.wikipedia.org/wiki/LanguageTool> (Дата обращения: 16.11.2020);
7. Яндекс. Спеллер. О сервисе - API. Руководство разработчика – [Электронный ресурс]:
<https://yandex.ru/dev/speller/doc/dg/concepts/speller-overview.html> (Дата обращения: 16.11.2020);
8. Android documentation – [Электронный ресурс]:
<https://developer.android.com/docs?hl=ru> (Дата обращения: 16.11.2020);
9. Apple Developer Documentation – [Электронный ресурс]:
<https://developer.apple.com/documentation/> (Дата обращения: 16.11.2020);
10. Java Virtual Machine – [Электронный ресурс]:
https://ru.wikipedia.org/wiki/Java_Virtual_Machine (Дата обращения: 23.12.2020);