МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра Прикладная математика и информатика
(наименование)

09.03.03 Прикладная информатика
(код и наименование направления подготовки / специальности)

Бизнес-информатика
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему Разработка информационной системы расшифровки анализов крови

Обучающийся М.Е. Кондратьев
(Инициалы Фамилия) (личная подпись)

Руководитель В.В. Дружинкин
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2022

Аннотация

Тема: «Разработка информационной системы расшифровки анализов крови».

Целью данной выпускной квалификационной работы является разработка информационной системы расшифровки анализов крови.

Работа состоит из введения, трех глав, заключения, списка использованной литературы и приложений.

В главе 1 происходит анализ области сдачи и расшифровки анализов крови. Составляются модели бизнес-процесса, производится сравнение аналогичных систем, а также производится формулирование задачи на разработку информационной системы.

В главе 2 происходит логическое моделирование информационной системы. Производится выбор технологии логического моделирования, и строятся диаграммы вариантов использования, классов и последовательности, а также проектируется база данных.

В главе3 происходит физическое моделирование информационной системы. Производится выбор архитектуры, технологий и подходов для разработки, проектируется физическая модель базы данных, реализуются мобильное и серверное приложения, а после проводится их тестирование.

В заключении происходит подведение результатов работы и формирование выводов.

Результатом выполнения бакалаврской работы является разработанная информационная система расшифровки анализов крови.

Бакалаврская работа состоит из пояснительной записки на 52 страницы, включая 21рисунок, 4 таблицы и список использованной литературы из 20 источников.

Abstract

The subject of the bachelor's thesis is «Development of an information system for the interpretation of blood tests».

The purpose of this bachelor's thesis is developing an information system for the interpretation of blood tests.

The whole work consists of an introduction, three chapters, a conclusion, a list of references and applications.

In chapter 1, an analysis of the field of interpretation of blood tests was made. Models of the business process have been compiled, similar systems have also been compared, and the task for the development of an information system has been formulated.

In chapter 2, the logical modeling of the information system was made. A choice of logical modeling technology is made and diagrams of use cases, classes and sequences are built, in addition a database was designed.

In chapter 3, the physical modeling of the information system was made. The choice of architecture, technologies and approaches for development is made, the physical model of the database is designed, mobile and server applications are implemented, and after that, they are all were tested.

In conclusion, the results of the work are summed up and all the related conclusions are formulated.

The result of the bachelor's thesis is the developed information system for the interpretation of blood tests.

Bachelor's thesis consists of an explanatory note on 52 pages, including 21 figures, 4 tables and a list of references from 20 sources.

Оглавление

Введение

В настоящее время вопрос здоровья стоит особенно остро. Различные факторы окружающей среды могут негативно влиять на здоровье человека. Одним из самых простых способов контроля за своим здоровьем является сдача анализов крови.

К сожалению, результаты анализов не всегда бывают очевидны простому человеку. Помимо этого, возникают проблемы с отслеживанием ДИНАМИКИ анализов.

Систематизация и интерпретация анализов крови позволит упростить процесс отслеживания показателей крови для своевременного обнаружения отклонений от норм.

Актуальность выпускной квалификационной работы состоит в необходимости повышения уровня осведомленности о возможных проблемах с организмом, а также повышения мотивации для обращения за квалифицированной медицинской помощью.

Для помощи в решении этой проблемы будет разработано мобильное приложение для расшифровки анализов крови.

В функции приложения будет входить возможность добавлять анализы крови для их хранения и удобного отслеживания динамики показателей, а также расшифровка анализов в зависимости от отклонений конкретных показателей.

Для управления этим приложением будет реализована система управления приложением.

Объектом выпускной квалификационной работы является процесс расшифровки анализов крови.

Предметом выпускной квалификационной работы является автоматизация процесса расшифровки анализов крови.

Целью выпускной квалификационной работы является разработка информационной системы расшифровки анализов крови.

Для достижения указанной цели поставлены следующие задачи: систематизировать знания о предметной области, проанализировать существующие решения, спроектировать базу данных, выбрать архитектуру, разработать информационную систему, провести тестирование компонентов системы.

В заключении подводятся итоги работы, информируются окончательные выводы по рассматриваемой теме.

Глава 1 Анализ предметной области автоматизации расшифровки анализов крови

1.1 Технико-экономическая характеристика процесса расшифровки анализов крови

Одним из самых простых, безопасных и информативных способов заметить нарушения в работе организма является исследование крови. Сдача крови для анализа –знакомая всем процедура, применяемая как для диагностики заболеваний, так и для их профилактики. Любые патологические процессы влияют на концентрацию показателей крови. Изменение отдельно взятого или нескольких показателей весьма специфично и характерно для различной группы заболеваний.

Проведение анализов крови происходит в лаборатории строго по стандартам. Главная задача лаборатории – обеспечить правильное проведение анализов. Согласно стандартам ГОСТ Р 53022.1-2008 и ГОСТ Р ИСО 15189-2015проведение лабораторных исследований разделено на 3 этапа [1] [2]:

- пре-аналитический (сбор биоматериала);
- аналитический (проведение анализа);
- пост-аналитический (выдача результатов).

После получения результатов исследования крови возникает одна из самых сложных задач – их расшифровка.Под расшифровкой анализов крови понимается интерпретация отклонений показателей от их референсных значений и отслеживание динамики изменения этих показателей с течением времени [3].

Показатель крови – это измеряемая характеристика, которая в случае отклонения от референсных значений может сигнализировать о патологических процессах, проходящих в организме человека [3].

Референсные значения – это пределы, в которых результаты исследований считаются нормой для здорового человека. Они могут быть как

конкретным числовым интервалом «от» и «до», так и ответом «положительно» или «отрицательно». Эти значения формируются на основе проведения специальных тестовых анализов среди здоровых людей. Именно поэтому эти интервалы не являются точными, а лишь позволяют обозначить границы из выборки у людей без каких-либо заболеваний [3].

1.2 Разработка модели бизнес-процесса расшифровки анализов крови «КАК ЕСТЬ»

Для определения целей использования информационной системы необходимо провести анализ бизнес-процессов, которые планируется автоматизировать. Нагляднее всего использовать методологию IDEF0[4].

0-й уровень методологии IDEF0 позволяет рассмотреть процесс расшифровки анализов крови. На построенной диаграмме (рисунок 1). отображен функциональный блок «Расшифровка анализов крови».

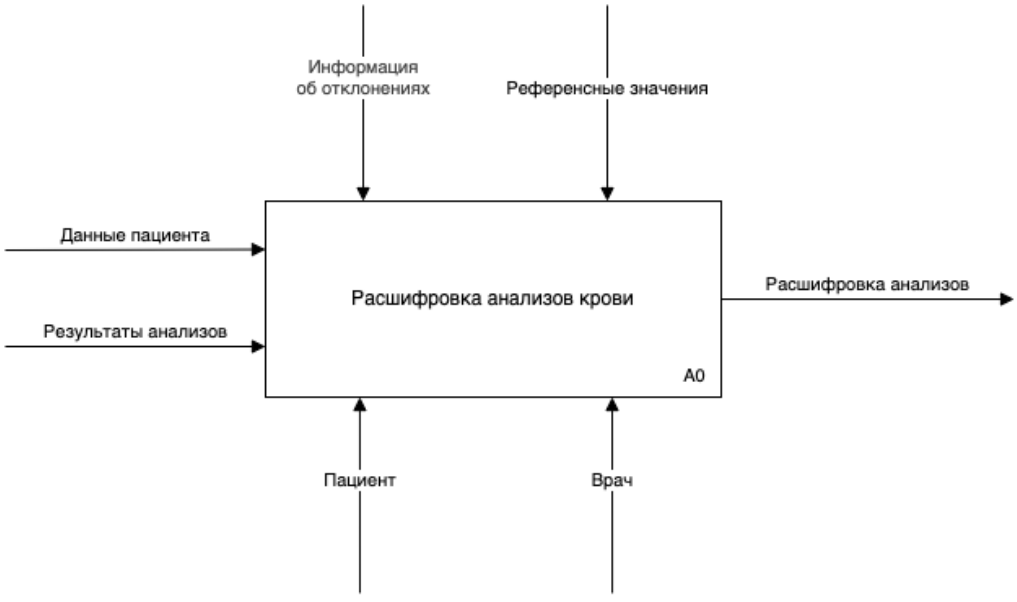

Рисунок 1. -Диаграмма процесса расшифровки анализов крови «КАК ЕСТЬ»

Из диаграммы выше видно, что бизнес-процесс на входе имеет данные пациента и результаты анализов. На выходе бизнес-процесса пациенту предоставляются расшифрованные результаты анализов.

Следующим шагом является декомпозиция бизнес-процесса верхнего уровня на бизнес-процессы, проходящие внутри него. Это позволит рассмотреть процесс наиболее полно и выявить узкие места, требующие автоматизации (рисунок 2) была произведена декомпозиция процесса нулевого уровня.

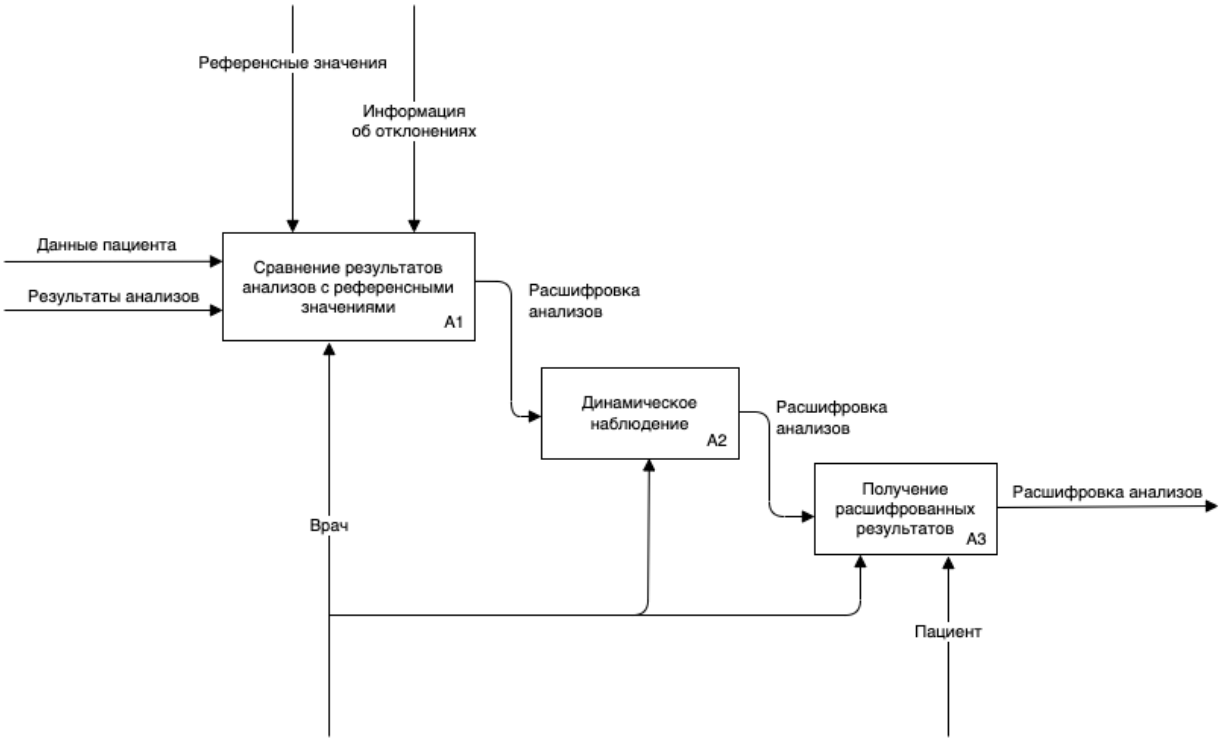

Рисунок 2 -Декомпозиция процесса расшифровки анализов крови «КАК ЕСТЬ»

Из диаграммы (рисунок 2) видно, что 0-й бизнес-процесс «Расшифровка анализов крови» состоит из трех под-процессов: «Сравнение результатов анализа с референсными значениями», «Динамическое наблюдение» и «Получение расшифрованных результатов».

В первом процессе врач производит сравнение показателей анализов с референсными интервалами для этих показателей, а также производит трактовку отклонений на основе информации об отклонениях. В качестве источника такой информации может являться справочник с интерпретацией отклонений.

Далее в процессе «Динамическое наблюдение» врач отслеживает показатели анализов в динамике. С помощью этого можно наблюдать за изменением показателей с течением времени сравнивая результаты анализов между собой при их наличии. Это помогает отслеживать состояние организма.

В последнем процессе «Получение расшифрованных результатов» пациенту предоставляются результаты анализов с указанием на показатели, у которых имеются отклонения, а также интерпретацию этих отклонений.

На основе анализа моделей бизнес-процесса расшифровки анализов крови можно прийти к выводу, что действия врача можно автоматизировать. Это поможет быстрее проводить расшифровку результатов анализов. Также информационная система поможет наглядно отслеживать динамику показателей, так как все анализы будут храниться в информационной системе. Это также поможет избежать утрату результатов анализов.

В итоге делается вывод, что автоматизированный вариант решения является целесообразным, так как способен упростить процедуру расшифровки и хранения анализов крови.

Для того, чтобы выяснить, какой именно функционал необходим в будущей информационной системе требуется сформировать к ней требования.

1.3 Формирование требований к новой технологии

Благодаря анализу бизнес-процесса расшифровки анализов крови можно составить список функциональных требований к информационной системе расшифровки анализов крови.

Для пациентов должно быть разработано мобильное приложение, а для врачей система управления контентом в виде веб-приложения.

Требования к мобильному приложению:

- регистрация и аутентификация пациентов;
- отображение общего списка всех анализов крови;
- добавление новых анализов;

– редактирование существующих анализов;
– просмотр деталей анализов;
– просмотр деталей конкретного показателя анализа;

Требования к системе управления:

– аутентификация врача;
– добавление анализов пациента;
– редактирование существующих анализов пациента;
– добавление и редактирование референсных значений;
– добавление и редактирование информации об отклонениях от референсных значений;
– добавление и редактирование единиц измерения показателей.

1.4Сравнительная характеристика существующих разработок

При создании информационной системы расшифровки анализов крови были изучены похожие решения, которые работают по схожему принципу.

Для сравнения выбранных решений были определены следующие критерии:

– регистрация и аутентификация пациентов;
– ручное добавление анализов;
– хранение результатов анализов;
– расшифровка отклонений в анализах;
– отслеживание динамики показателей;
– стабильность системы;
– простота интерфейса;
– ограничения функционала;
– стоимость использования.

Для сравнения были выбраны наиболее популярные решения на рынке. У каждого из них есть свои плюсы и минусы. Необходимо рассказать о каждом в отдельности.

«Ornament» — это мобильное приложение, которое помогает следить за здоровьем на основе анализов крови. Данное приложение позволяет не только отслеживать динамику анализов, но и следить за влиянием анализов на весь организм.Особенностями приложения можно выделить монитор здоровья, режим «Беременность», магазин анализов, а также оцифровку анализов. Недостатками приложения (рисунок3) являются высокая стоимость подписки и ограниченная функциональность бесплатной версии, а также высокая частота сбоев приложения по отзывам из магазинов приложений [6].

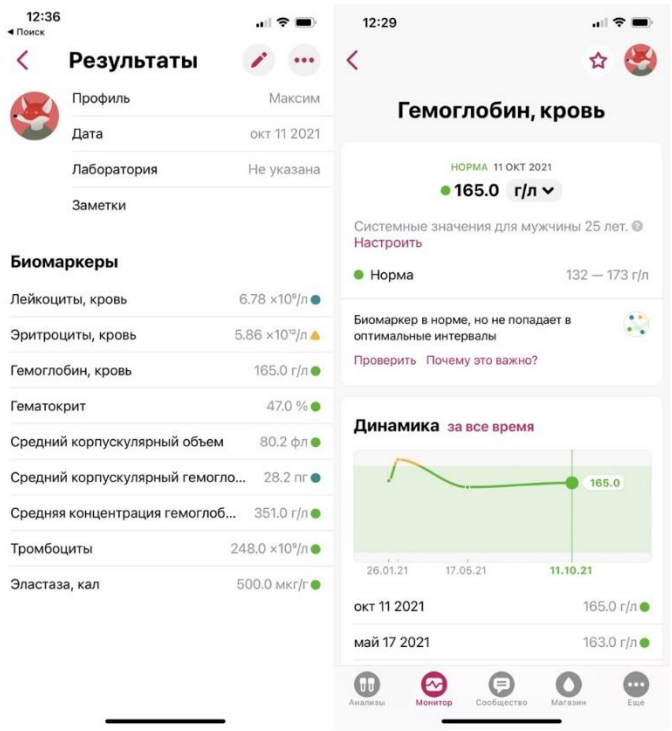

Рисунок 3 - Мобильное приложение «Ornament»

«Biogenom»— это электронная медицинская карта, (рисунок4) которая позволяет хранить медицинские данные, документы и анализы в одном месте. Особенностями продукта является консультация врача онлайн, возможность хранения различных документов, семейный профиль. Недостатками

приложения являются высокая стоимость подписки, ограниченная функциональность бесплатной версии, громоздкий интерфейс, отсутствие возможности добавления анализов в ручном режиме и нестабильность системы [7].

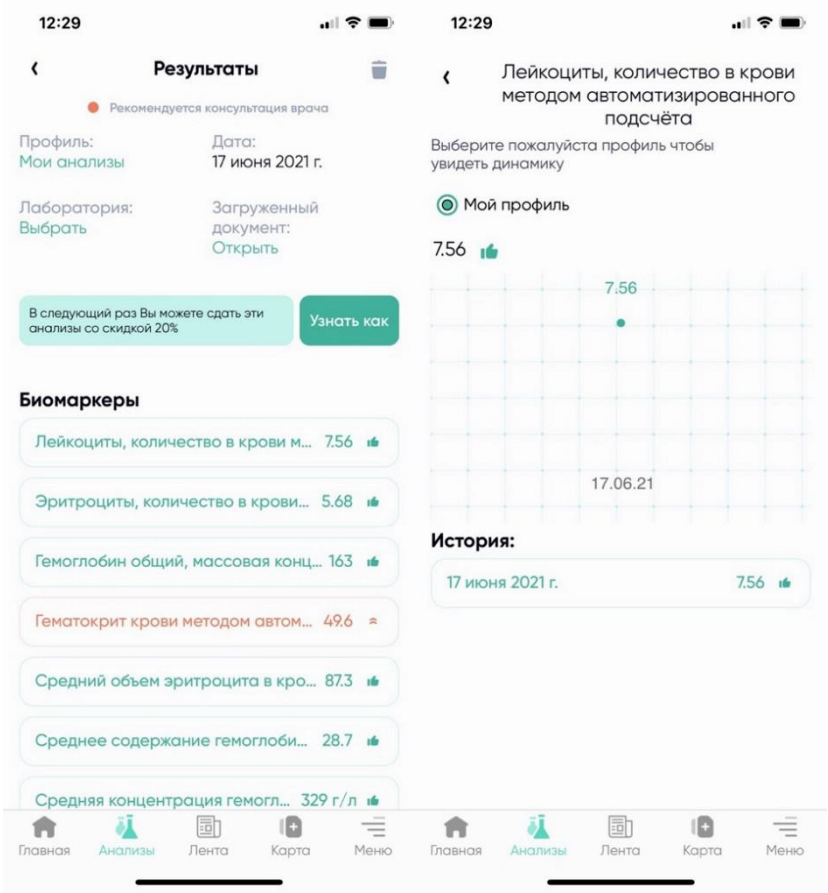

Рисунок 4 - Мобильное приложение «Biogenom»

Необходимо сравнить данные сервисы друг с другом по критериям, которые мы определили выше. Сравнительная характеристика функциональных показателей представлена в таблице 1

Таблица 1 – Таблица сравнение аналогов

Критерии	Ornament	Biogenom
Регистрация и аутентификация пациентов	+	+
Ручное добавление анализов	+	-
Хранение результатов анализов	+	+
Расшифровка отклонений в анализах	+	+

Отслеживание динамики показателей	+	+
Стабильность системы	-	-
Простота интерфейса	+	-
Ограничения функционала	-	-
Стоимость использования	-	-

На основе полученного сравнительного анализа приложений «Ornament» и «Biogenom» в таблице 1.1 можно сделать вывод, что каждый из этих сервисов в какой-то мере удовлетворяет критериям сравнения, но несмотря на это имеет и свои недостатки. Поэтому есть необходимость разработать информационную систему расшифровки анализов крови.

1.5 Постановка задачи на разработку информационной системы расшифровки анализов крови

На основе проведенного анализа предметной области и сравнительном анализе приложений необходимо поставить задачу на разработку информационной системы расшифровки анализов крови. Это необходимо в первую очередь для конкретизации аспектов реализации.

Информационная система создается со следующими целями:

- автоматизации расшифровки отклонений анализов крови;
- автоматизации отслеживания динамики показателей;

На основе этого составлен список задач для выполнения:

- провести логическое моделирование информационной системы;
- выбрать архитектуру информационной системы;
- провести анализ технологий для реализации;
- реализовать компоненты информационной системы: мобильное приложение и серверное приложение;
- провести тестирование вышеуказанных компонентов.

Исходя из вышеописанного, определены цели и задачи на разработку информационной системы расшифровки анализов крови.

1.6 Разработка модели бизнес-процесса «КАК ДОЛЖНО БЫТЬ»

На основе модели бизнес-процесса расшифровки анализов крови «КАК ЕСТЬ» и ее дальнейшего анализа разработана модель «КАК ДОЛЖНО БЫТЬ» (рисунок 5), которая позволяет наглядно увидеть результат реализации информационной системы [4].

Рисунок 5 - Диаграмма процесса расшифровки анализов крови «КАК ДОЛЖНО БЫТЬ»

На (рисунке .6) изображена декомпозиция диаграммы бизнес-процесса расшифровки анализов крови «КАК ДОЛЖНО БЫТЬ» для более наглядного представления бизнес-процессов, проходящих внутри.

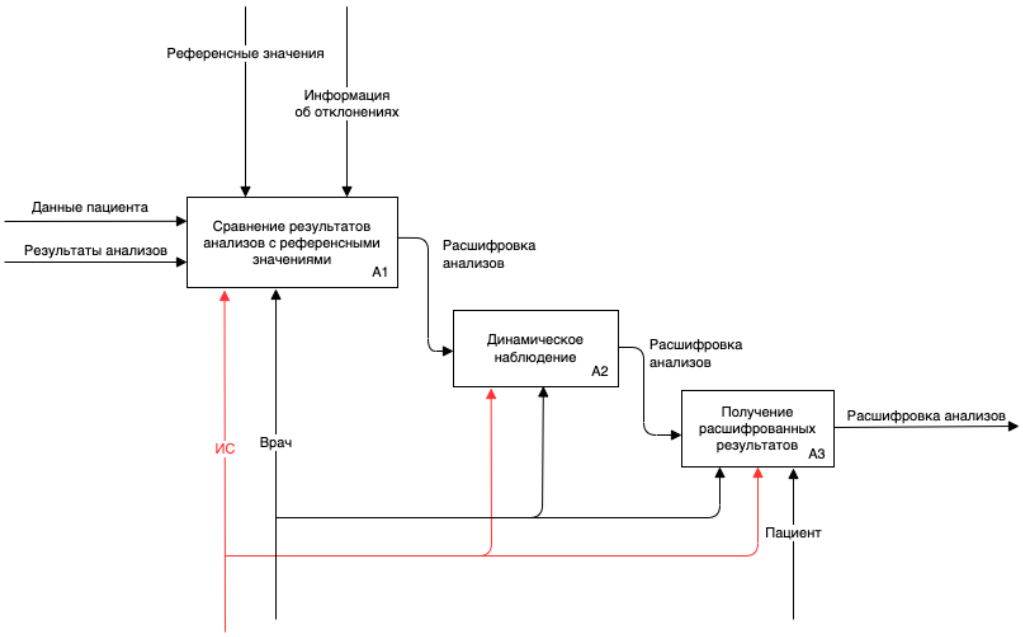

Рисунок 6 - Декомпозиция процесса расшифровки анализов крови «КАК ДОЛЖНО БЫТЬ»

Из чего видно, что после внедрения ИС будут усовершенствованы все под-процессы. При использовании информационной системы будет получена расшифровка анализов крови и добавлена возможность отслеживать динамику показателей с течением времени.

Вывод по первой главе

В данной главе был проведен анализ предметной области расшифровки анализов крови, было проведено моделирование бизнес-процесса с помощью диаграммы «КАК ЕСТЬ», составлены требования к информационной системе, проведен анализ похожих решений. Все вышеописанные действия позволили определить необходимость разработки информационной системы расшифровки анализов крови. После этого были определены цели и задачи на разработку информационной системы и была разработана диаграмма бизнес-процесса расшифровки анализов крови «КАК ДОЛЖНО БЫТЬ».

Глава 2Проектированиеинформационной системы расшифровки анализов крови

2.1 Выбор технологии логического моделирования информационной системы расшифровки анализов крови

После анализа предметной области следует провести логическое моделирование информационной системы расшифровки анализов крови.

Для логического моделирования принято решение использовать унифицированный язык для визуального моделирования UML(UnifiedModelingLanguage). Этот язык включает в себя большой набор диаграмм для описания всех аспектов разрабатываемой информационной системы.

Основные элементы UML:

- Диаграммы – это блоки логической модели для группировки сущностей и связей;
- Сущности – это объекты модели;
- Связи – это соединения между сущностями.

Для логического моделирования чаще всего используются диаграммы вариантов использования, последовательности и классов [5].

Для наглядного представления требований, которые должны быть реализованы в информационной системе, применяется диаграмма вариантов использования. В этой диаграмме отображаются актеры и прецеденты. При помощи прецедентов формируется требуемое поведение информационной системы.

Для отображения последовательности взаимодействия актеров в рамках определенного прецедента используется диаграмма последовательности. В этой диаграмме на временной оси отображается жизненный цикл какого-либо определенного объекта и взаимодействие актеров информационной системы.

Для иллюстрации структуры информационной системы используется диаграмма классов. Она состоит из классов, атрибутов и методов, а также из отношений между объектами.

2.2 Построение логической модели информационной системы расшифровки анализов крови

Для более наглядного представления, какие актеры задействованы в информационной системе и их взаимодействия между собой построим диаграмму вариантов использования.

Выделим следующих актеров:

– Пациент – человек, который пользуется мобильным приложением для управления своими анализами;
– Врач – человек, который вносит данные в систему управления контентом.

В таблице 2 представлено описание прецедентов, которые могут совершать вышеуказанные актеры.

Таблица 2 - Описание прецедентов

Прецедент	Описание
Регистрация и аутентификация	Действие пациента и врача для регистрации и аутентификации в системе.
Добавление анализов	Действие пациента и врача для добавления анализов в БД
Редактирование анализов	Действие пациентаи врача для внесения изменений в уже существующий анализ в БД
Просмотр деталей анализов	Действие пациента для ознакомления с детальной информации о значениях показателей анализов
Просмотр деталей конкретного показателя анализа	Действие пациента для ознакомления с детальной информацией о конкретном показателе анализа, его истории и причине возможных отклонений
Добавление референсных значений	Действие врача для добавления референсных интервалов для конкретного показателя анализа в БД
Добавление единиц измерения	Действие врача для добавления единицы измерения для показателя анализа в БД

Редактирование единиц измерения	Действие врача для внесения изменений в уже существующую единицу измерения в БД

Продолжение таблицы 2

Прецедент	Описание
Добавление отклонений от референсных значений	Действие врача для добавления причин отклонения конкретного показателя анализа от его референсного значения в БД
Добавление биомаркеров	Действие врача для добавления возможных показателей анализов в БД
Редактирование биомаркеров	Действие врача для редактирования уже существующего возможного показателя анализа в БД

На основе таблицы прецедентов выполнено построение диаграммы (рисунок 7) вариантов использования

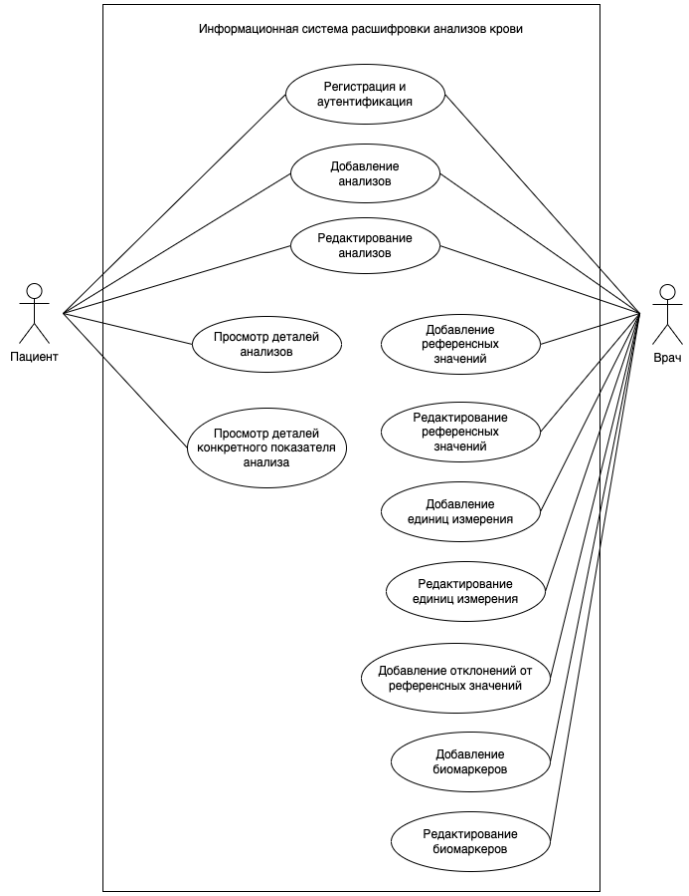

Рисунок 7 - Диаграмма вариантов использования

Диаграмма вариантов использования отображает основной функционал информационной системы.

Диаграмма последовательности (рисунок 8) добавления анализа пациента в информационную систему.

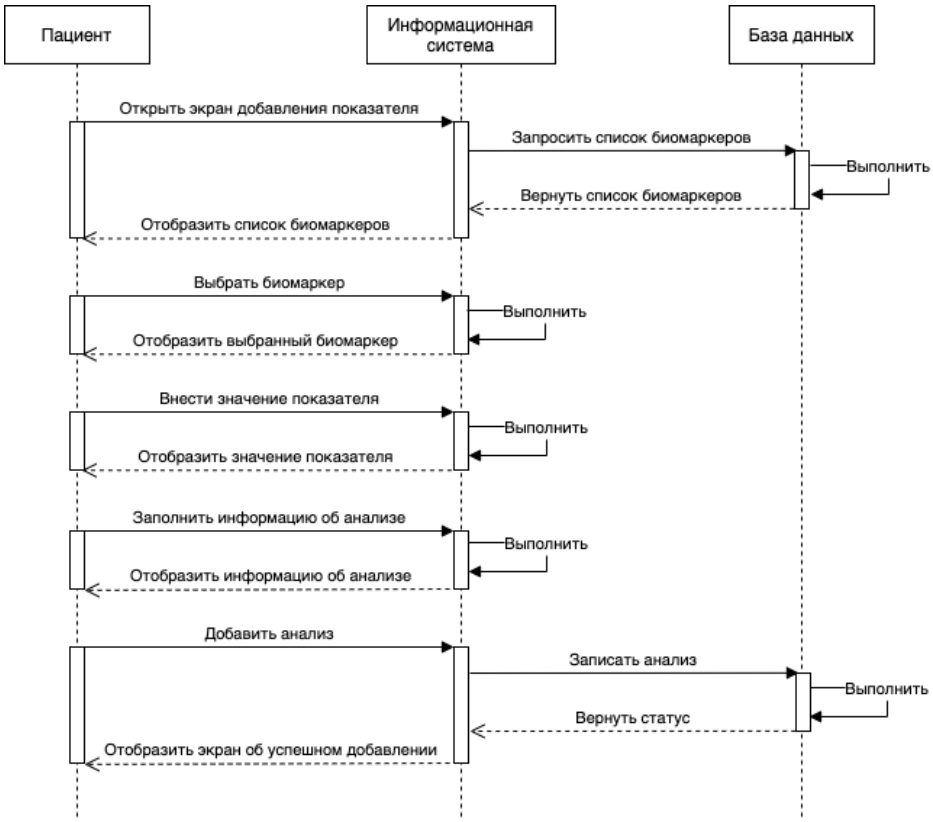

Рисунок 8 - Диаграмма последовательности добавления анализа пациентом в информационную систему

Пациент открывает экран с добавлением показателей анализа, после чего ему отображается список со всеми доступными показателями. В этот момент выполняется запрос в базу данных для получения этого списка. После чего список возвращается пациенту.

Пациент выбирает необходимый показатель из списка, после чего он отображается в списке добавленных показателей анализа.

Пациент вводит значение выбранного показателя, после чего это значение отображается возле показателя.

Пациент заполняет информацию об анализе, такую как название, дату проведения и комментарий. Эта информация отображается пациенту.

Пациент добавляет анализ. В это время выполняется запрос в базу данных для записи анализа, после чего возвращается статус анализа и пациенту отображается экран об успешном добавлении анализа.

На основе вышеуказанных диаграмм составлена диаграмма классов информационной системы (рисунок 9), в которой:

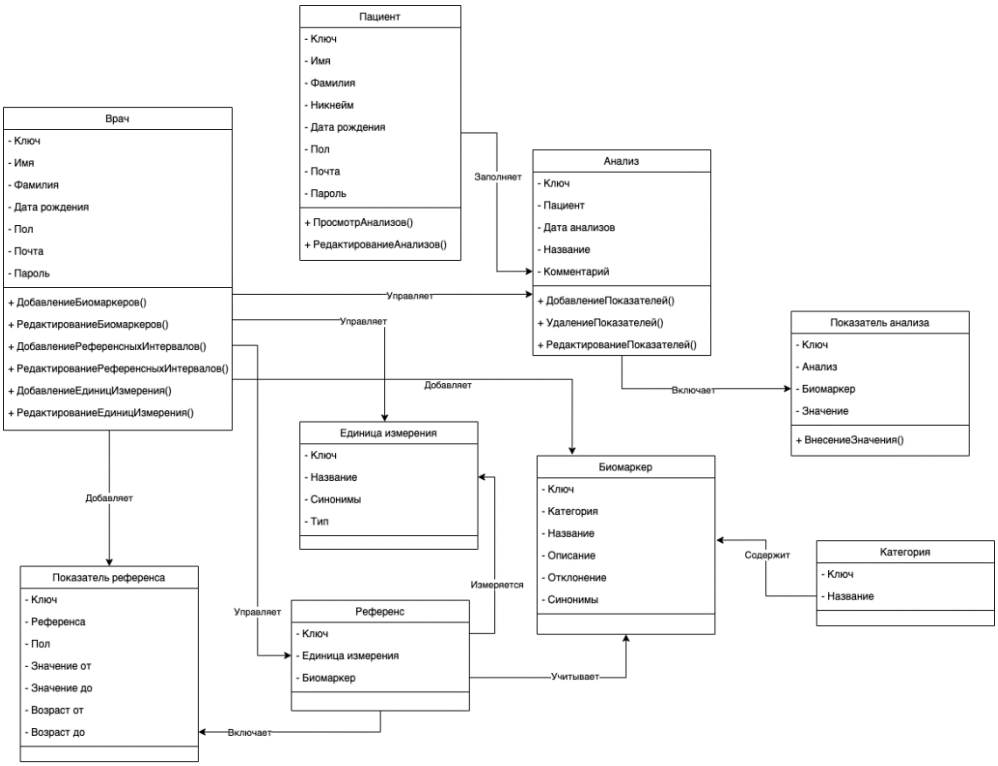

Рисунок 9 - Диаграмма классов информационной системы расшифровки анализов крови

– «Пациент» – класс пациентов-пользователей системы;
– «Анализ» – класс анализов, является совокупностью показателей;
– «Показатель анализов» – класс показателей анализов, являющийся отдельной характеристикой анализа;
– «Врач» – класс врачей-пользователей системы;
– «Биомаркер» – класс возможного показателя анализа;
– «Категория» – класс категории конкретного показателя;

– «Референс»– класс совокупности референсных интервалов для конкретного показателя;
– «Единица измерения» – класс единицы измерения конкретного показателя;
– «Показатель референса» – класс показателей референса, являющийся отдельным интервалом.

Построенные диаграммы используются для дальнейшего создания логической модели базы данных.

2.3 Проектирование базы данных информационной системы расшифровки анализов крови

Для проектирования и прототипирования базы данных используется логическая модель. Логическая модель (рисунок 10) создается для наглядного представления базы данных в виде сущностей и их взаимосвязей между собой.

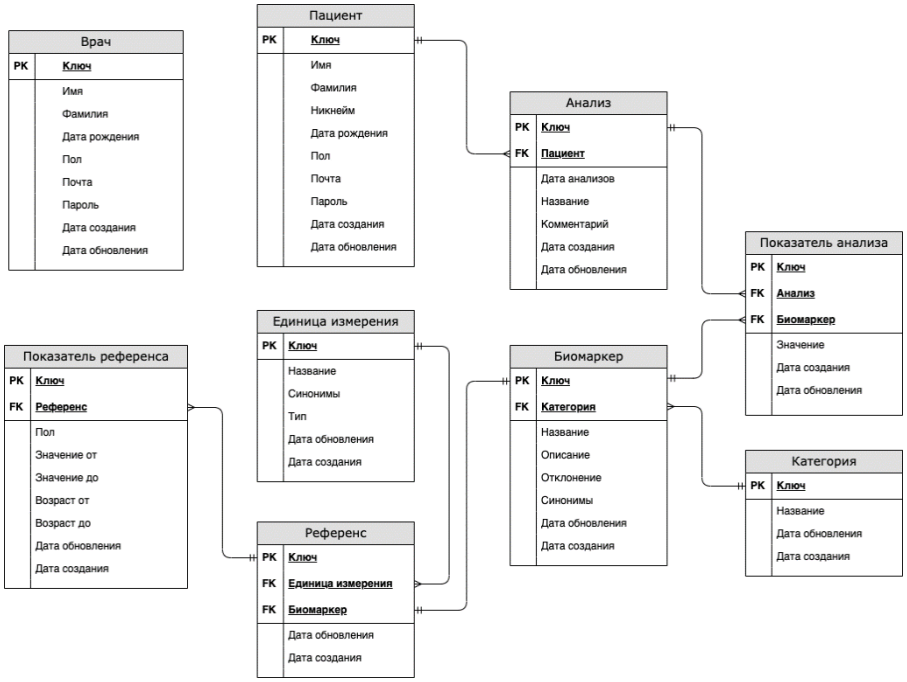

Рисунок 10 - Логическая модель базы данных

Логическая модель состоит из 9 сущностей. У каждой сущности есть свой уникальный ключ-идентификатор, по которому осуществляется связь с остальными сущностями.

Сущность «Врач» является независимой и не связана ни с какой сущностью.

Сущность «Пациент» связана с сущностью «Анализ» связью «один-ко-многим», так как у одного пациента может быть много различных анализов.

Сущность «Анализ» связана с сущностью «Показатель анализа» связью «один-ко-многим», так как в рамках одного проведенного анализа может быть определено несколько показателей крови.

Сущность «Категория» связана с сущность «Биомаркер» связью «один-ко-многим», так как у одной категории может быть несколько биомаркеров.

Сущность «Биомаркер» связана с сущностью «Показатель анализа» связью «один-ко-многим», так как один биомаркер может иметь несколько показателей крови для разных анализов. Так же сущность «Биомаркер» связана с сущностью «Референс» связью «один-к-одному», так как у одного биомаркера может быть только один референс.

Сущность «Единица измерения» связана с сущностью «Референс» связью «один-ко-многим», так как каждая единица измерения может быть использована у нескольких референсов.

Сущность«Референс» связана с сущностью «Показатель референса» связью «один-ко-многим», так как у каждого референса может быть несколько референсных интервалов для каждого пола и возраста.

Вывод по второй главе

В данной главе была выбрана технология логического моделирования информационной системы расшифровки анализов крови, была составлена таблица прецедентов и построены диаграммы вариантов использования, последовательности и классов. После этого была разработана логическая модель базы данных.

Глава 3Реализацияинформационной системы расшифровки анализов крови

3.1 Выбор архитектуры информационной системы расшифровки анализов крови

Для реализации информационной системы принято решение использовать трёхзвенную клиент (рисунок 11) - серверную архитектуру [10]

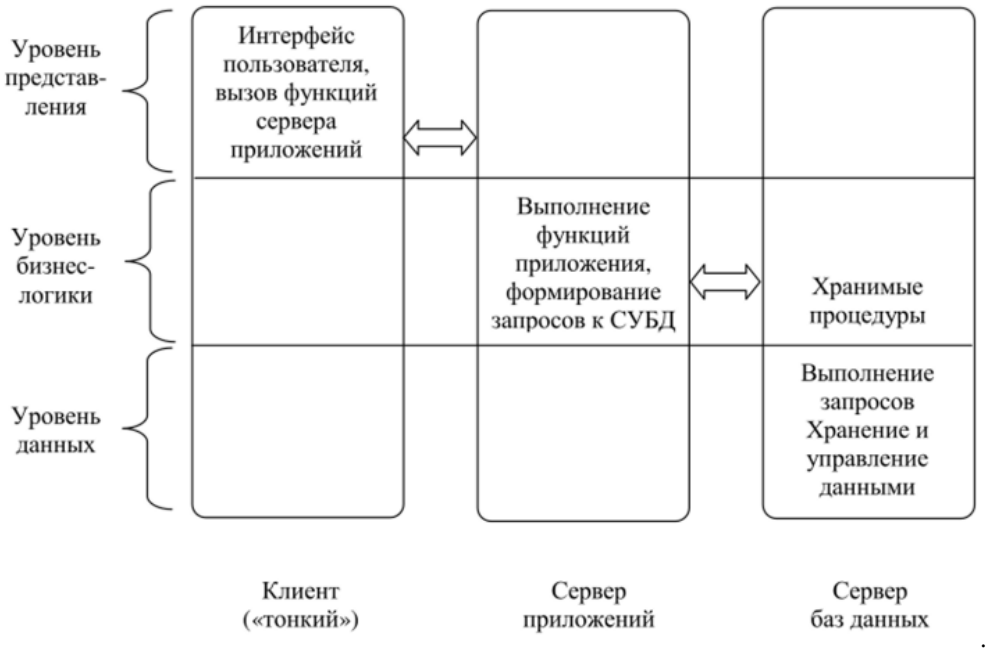

Рисунок 11–Трёхзвенная архитектура «Клиент-Сервер»

Уровень представления –это самый верхний слой, представляющий из себя интерфейс пользователя. С его помощью осуществляется взаимодействие с пользователем и вызов функций сервера приложений.

Уровень бизнес-логики – это координирующий слой между уровнем представления и уровнем данных. Он служит для обработки команд, выполнения функций приложения и различных вычислений, а также он отвечает за формирования запросов к системе управления базой данных.

Уровень данных – это самый нижний слой, на котором происходит выполнение запросов от уровня бизнес-логики. Он отвечает за хранение и управления данными. Информация из этого уровня возвращается на запросы от уровня бизнес-логики для обработки и последующей передаче конечному пользователю.

3.2 Выбор технологии разработки программного обеспечения информационной системы расшифровки анализов крови

На основании выбора трёхзвенной клиент-серверной архитектуры необходимо выбрать технологии для реализации клиентского приложения, серверного приложения и системы управления базой данных.

В качестве клиентской части выбрано мобильное приложение.

Для разработки мобильных приложений существует 2 наиболее популярных способа:

– Разработка нативных приложений;

– Разработка кроссплатформенных приложений.

Разработка нативных приложений –это принцип разработки мобильных приложений под конкретную платформу. Самыми популярными платформами на данный момент являются Androidи iOS. Для разработки под Androidиспользуются языки программирования JavaиKotlin. Для разработки под iOSиспользуются языки программирования Objective-Си Swift.

Разработка кроссплатформенных приложений – это принцип разработки мобильных приложений с использованием библиотеки фреймворков, которые позволяют исполнять единожды написанный код на нескольких платформах. Примером такого фреймворка является ReactNative.

ReactNative – это кроссплатформенный фреймворк для разработки мобильных приложений на языке JavaScript. Он выступает связующим звеном, позволяя писать код на одном языке, который в последствии взаимодействует с нативными модулями конкретной платформы [18].

Для реализации мобильного приложения выбран фреймворк ReactNative. Он позволяет заметно упростить и сократить время на разработку путем использования языка программирования JavaScript [16].

Что бы сократить количество ошибок в коде и сделать его более читаемым принято решение использовать язык программирования со статической типизациейTypeScript. Код, написанный на этом языке, в конечном итоге преобразовывается в код на JavaScript [17].

После этого необходимо выбрать технологию для реализации серверной части. Так как для реализации клиентской части используется язык программированияJavaScript, для последующего упрощения разработки принято решение использовать платформу Node.js.

Node.js – это платформа для исполнения JavaScriptкода на сервере с помощью движка V8, который преобразует код на JavaScriptв более быстрый низкоуровневый код.

Для управления контентом принято решение использовать систему управления контентомStrapiпо причине ее многофункциональности.

Strapi – это фреймворк с открытым исходным кодом для управления контентом, работающий на Node.js. Помимо упрощения работы с данными путем предоставления удобной панели администратора, этот фреймворк позволяет быстро разрабатывать APIдля взаимодействия между клиентом и сервером [19].

Следующим шагом производится выбор способа обмена данными между мобильным приложением и сервером. В Strapiесть поддержка 2-ух способов:

– RESTAPI;

– GraphQL.

RESTAPI – это способ для взаимодействия между клиентом и сервером с помощью протокола HTTP. Для обмена данными клиент посылает запросы на определенные конечные точки на сервер, а сервер возвращает данные в формате JSON. Для взаимодействия используются следующиеHTTPметоды:

– GET – для получения данных;
– POST – для создания данных;
– PUT – для редактирования данных;
– DELETE – для удаления данных.

GraphQL – это язык для запросов данных и манипулирования ими. В GraphQLклиент описывает какую операцию он хочет выполнить и какие данным ему нужны с помощью схемы, а после посылает запросы на единственную конечную точку. Сервер в свою очередь возвращает данные строго опираясь на полученную схему от клиента. Этот принцип, в отличии от RESTAPI, позволяет не запрашивать данные, которые не нужны для клиента в данный момент, тем самым сокращая время выполнения запроса и размер возвращаемых данных [20].

Помимо этого, вStrapiесть «песочница» для тестирования и отладки GraphQLзапросов, которая так же является документацией.

После сравнения этих технологий для взаимодействия между клиентом и сервером выбрана технологияGraphQL [9].

3.3 Выбор СУБД для информационной системы расшифровки анализов крови

Следующим шагом необходимо выбрать СУБД для информационной системы. Существует 2 типа СУБД:

– SQL;
– NoSQL.

SQL – это реляционная база данных, где все данные хранятся в виде таблиц и находятся в строгой взаимосвязи друг с другом. В таблицах есть столбцы и строки. Каждый столбец является полем с указанным ему типом данных, а каждая строка является отдельной записью.

NoSQL – это не реляционная база данных, где все данные хранятся без взаимосвязей друг с другом. Вместо таблиц, в отличии от реляционных баз данных, данные внутри такой БД хранятся в виде документов.

Поскольку Strapi в значительной степени является реляционно-ориентированным фреймворком, использованиеSQLбазы данных является более целесообразным для получения всех преимуществ в производительности.

Разработчиками этой системы рекомендуется использовать СУБД PostgreSQL [19]. У этой системы управления базами данных есть следующие преимущества:

- Открытый исходный код;
- Долгая история;
- Частые обновления;
- Растущая популярность;
- Большое разнообразие форматов данных;
- Неограниченный размер базы данных;
- Высокая оценка пользователей.

На основе этого принято решение использовать СУБД PostgreSQL.

3.4 Разработка физической модели базы данных для информационной системы расшифровки анализов крови

Следующим шагом необходимо разработать физическую модель базы данных, которая будет использована в информационной системе (рисунок 12). Модель БД разработана с помощью языка DBML (приложение А) с учетом выбранной СУБД PostgreSQLи фреймворка Strapi.

DBML (DatabaseMarkupLanguage) – это язык, предназначенный для определения и документирования схем и структур базы данных. Онразработан, чтобы быть простым, последовательным и легко читаемым.

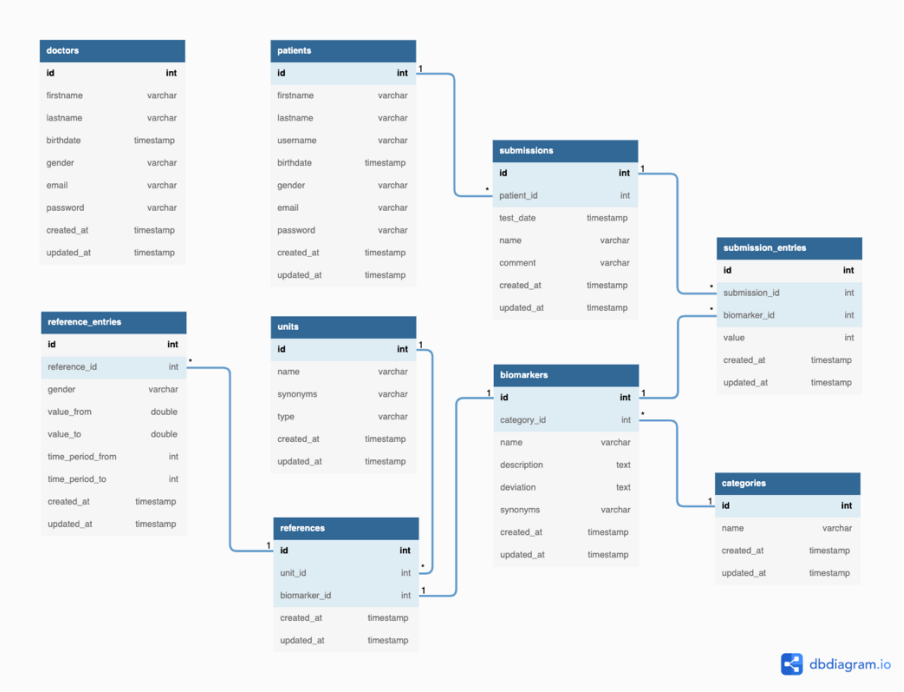

Рисунок 12 - Физическая модель базы данных информационной системы

Физическая модель данных состоит из следующих сущностей: «doctors»,«patients», «categories», «biomarkers», «units», «references», «submissions», «submission_entries», «reference_entries».

3.5 Разработка информационной системы расшифровки анализов крови

3.5.1 Разработка серверной части и базы данных

Серверная часть информационной системы реализована с помощью фреймворка Strapi, который работает на Node.js. Этот фреймворк включает панель администратора для управления контентом, систему для проектирования архитектуры контента, плагины для расширения функционала.Развертывание этого фреймворка происходит всего в одну команду. После того, как установщик закончил работу, в указанной при развертывании директории появилось сгенерированное приложение. Первым шагом нужно настроить подключение к базе данных. Это выполняется с

помощью специального конфигурационного файла, в котором указываются все необходимые переменные для подключения [12] (приложение Б).

После этого необходимо создать модели в базе данных. В Strapiесть 2 варианта создания моделей [13]:

– файл с конфигурацией;

– визуальный конструктор.

Удобнее делать это с помощью визуального конструктора. Задается название модели, после создаются все необходимые поля, указываются их типы и прочие параметры, а после сохраняется в системе (рисунок 13).

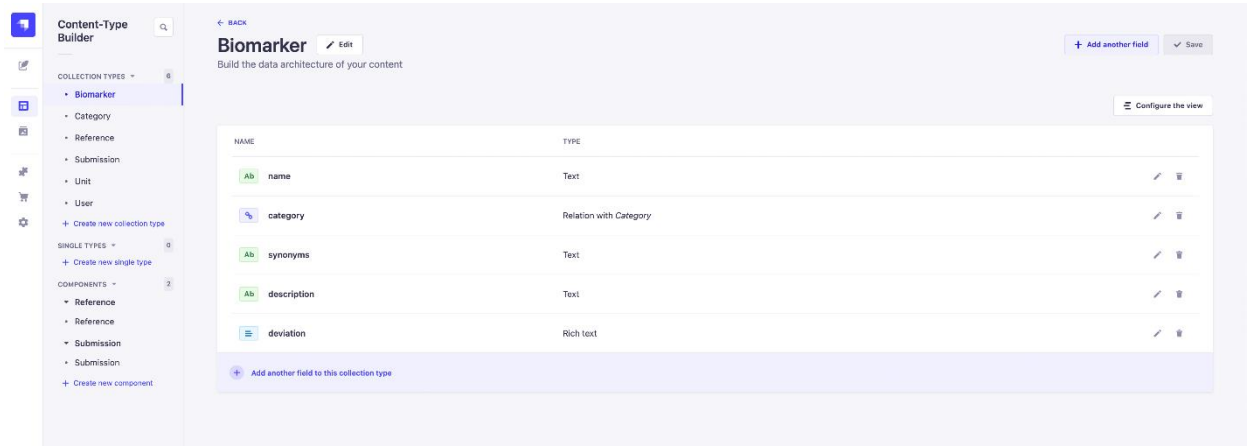

Рисунок 13 - Модель сущности в Strapi

Фреймворк на основе этого модифицирует файл с конфигурацией для каждой модели (приложение В) и генерирует точки доступа к API.

Для поддержки GraphQLнеобходимо установить специальный плагин [8]. Это так же выполняется с помощью одной консольной команды. После этого нам становится доступна «песочница», в которой можно отправлять GraphQLзапросы (рисунок 14).

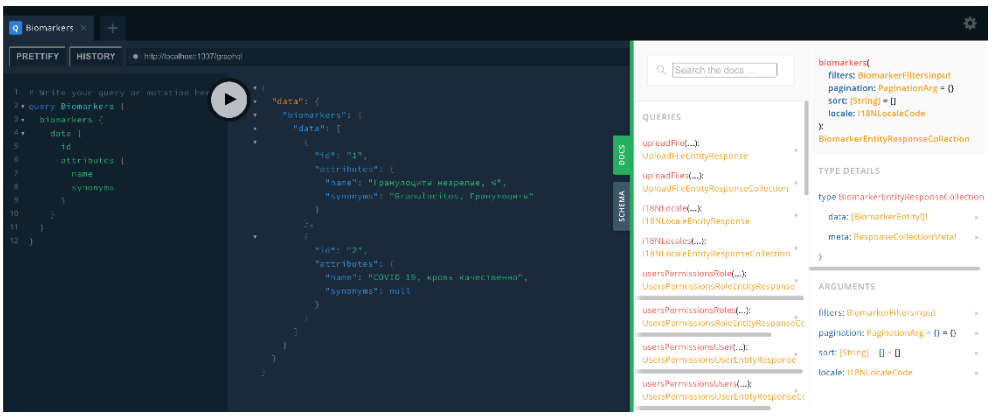

Рисунок 14 - «Песочница» и документация дляGraphQLзапросов

Так же здесь есть доступ к документации по каждому GraphQLзапросу [11].

3.5.2Разработка клиентской части

Для реализации мобильного приложения используется фреймворк ReactNative. Для создания базовой структуры проекта используется утилита ReactNativeCLI [14]. После установки всех необходимых зависимостей создается базовая структура файлов и папок. После этого необходимо продумать структуру файлов и папок для хранения исходного кодапроекта (рисунок 15).

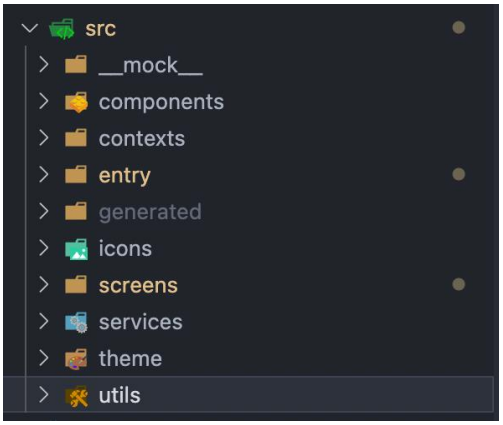

Рисунок 15 - Структура файлов и папок мобильного приложения

Весь исходный код проекта находится в папке src.

- «__mock__» – папка, в которой находятся тестовые данные, необходимые для отладки приложения;
- «components» – папка, в которой хранятся компоненты приложения, которые могут быть использованы в нескольких местах приложения;
- «contexts» – папка, в которой хранятся React-контексты приложения. Контексты нужны для обмена данными между компонентами;
- «entry» – папка, в которой находятся файлы, с которых начинается инициализация приложения. Здесь задается навигация и роутинг между экранами приложения;
- «generated» – папка, в которой находятся автоматически сгенерированные файлы приложения. Например, сгенерированные типы данных TypeScriptна основе GraphQL схемы;
- «icons» – папка, в которой хранятся SVGиконки приложения;
- «screens» – папка, в которой хранятся экраны всего приложения;
- «services» – папка, в которой хранятся сервисные файлы;
- «theme» – папка, в которой находятся конфигурационные файлы для стилизации приложения;
- «utils» – папка, в которой находятся файлы со вспомогательными функциями. Например, для форматирования дат.

Для подключения к серверу используется библиотека ApolloClient. Эта библиотека позволяет быстро осуществить подключение к конечной точке на сервере и выполнять запросы с помощью всего лишь нескольких строк. Всю основную рутинную работу библиотека берет на себя, отвечая за авторизацию, состояние запросов, обработку ошибок, кэширование данных [11] (рисунок 16)

```
import { ApolloClient, InMemoryCache, createHttpLink } from "@apollo/client";
import { setContext } from "@apollo/client/link/context";
import AsyncStorage from "@react-native-community/async-storage";

const httpLink = createHttpLink({
  uri: "http://localhost:4000/graphql",
});

const authLink = setContext(async (_, { headers }) => {
  const token = await AsyncStorage.getItem("accessToken");
  return {
    headers: {
      ...headers,
      authorization: token ? `Bearer ${token}` : null,
    },
  };
});

export const client = new ApolloClient({
  link: authLink.concat(httpLink),
  cache: new InMemoryCache(),
});
```

Рисунок 16 - Инициализация библиотеки ApolloClientдля подключения к GraphQLсерверу

Для того, чтобы по максимуму использовать все преимущества языка программирования TypeScript, установлена библиотека @graphql-codegen/cli, которая позволяет генерировать TypeScript типы на основе схемы данных GraphQL, которая находится на сервере [15]. Подключение и настройка библиотеки осуществляется в специальном файле с конфигурацией (приложение Г). При первом подключении эта библиотека получает структуру данных и запросов, а после создает файл со всеми типами данных и вспомогательными функциями для выполнения этих запросов (рисунок 17).

При каждом обновлении структуры на сервере типы генерируются заново, что довольно удобно. Это позволяет использовать автозаполнение во всех местах, где мы выполняем запросы к серверу, а также позволяет исключить ошибки возможного использования неподходящих функций и методов.

```ts
/**
 * __useGetAnalyseListQuery__
 *
 * To run a query within a React component, call `useGetAnalyseListQuery` and pass it any options that fit your needs.
 * When your component renders, `useGetAnalyseListQuery` returns an object from Apollo Client that contains loading, error, and data properties
 * you can use to render your UI.
 *
 * @param baseOptions options that will be passed into the query, supported options are listed on: https://www.apollographql.com/docs/react/api/react-hooks/#options;
 *
 * @example
 * const { data, loading, error } = useGetAnalyseListQuery({
 *   variables: {
 *   },
 * });
 */
export function useGetAnalyseListQuery(baseOptions?: Apollo.QueryHookOptions<GetAnalyseListQuery, GetAnalyseListQueryVariables>) {
        const options = {...defaultOptions, ...baseOptions}
        return Apollo.useQuery<GetAnalyseListQuery, GetAnalyseListQueryVariables>(GetAnalyseListDocument, options);
      }
export function useGetAnalyseListLazyQuery(baseOptions?: Apollo.LazyQueryHookOptions<GetAnalyseListQuery, GetAnalyseListQueryVariables>) {
          const options = {...defaultOptions, ...baseOptions}
          return Apollo.useLazyQuery<GetAnalyseListQuery, GetAnalyseListQueryVariables>(GetAnalyseListDocument, options);
        }
export type GetAnalyseListQueryHookResult = ReturnType<typeof useGetAnalyseListQuery>;
export type GetAnalyseListLazyQueryHookResult = ReturnType<typeof useGetAnalyseListLazyQuery>;
export type GetAnalyseListQueryResult = Apollo.QueryResult<GetAnalyseListQuery, GetAnalyseListQueryVariables>;
```

Рисунок 17 - Фрагмент сгенерированного кода на основе GraphQLсхемы

3.6 Описание функциональности информационной системы расшифровки анализов крови

3.6.1 Описание функциональности серверного приложения

Рассмотрим серверное приложение со страницы добавления сущности (рисунок 18).

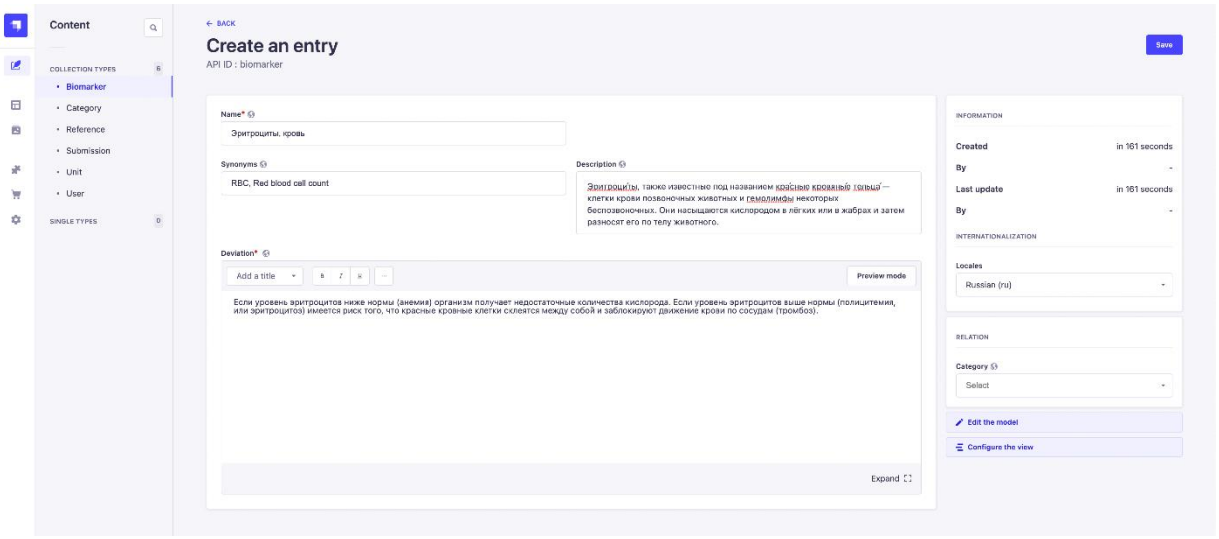

Рисунок 18 - Страница добавления сущности в Strapi

На странице добавления любой сущности в панели администратора находятся поля, которые указаны в модели. Система сама настраивает формат поля в зависимости от указанного типа. После добавления сущность попадает в общий список (рисунок 19). На этой странице можно настраивать вид отображения сущностей в таблице, а также фильтровать и выполнять поиск. У каждой сущности есть список действий: редактирование, клонирование и удаление.

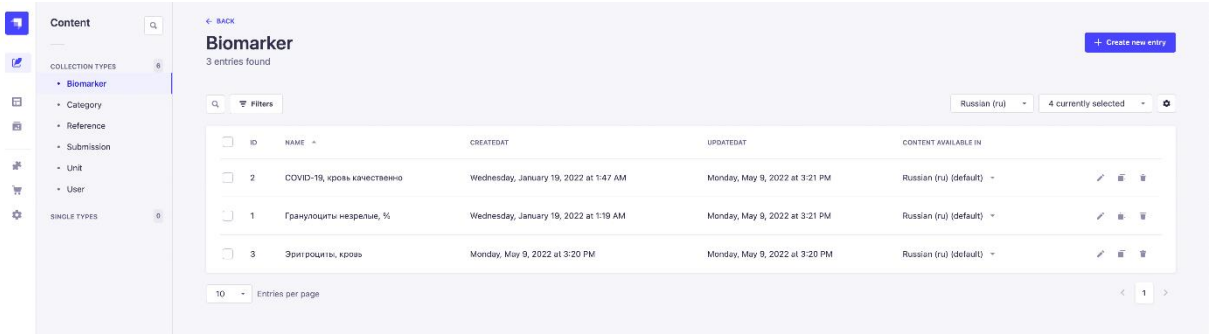

Рисунок 19 - Страница просмотра списка сущностей

По такому принципу происходит управление каждой сущностью в базе данных.

3.6.2Описание функциональности клиентского приложения

Рассмотрим мобильное приложение с главного экрана (рисунок 20).

При переходе на экран деталей анализа, помимо деталей анализа, можно увидеть каждый показатель в отдельности. У показателя есть его название и текущее значение. С помощью шкалы показаны референсные интервалы как минимальное и максимальное значение, а также текущее значение относительно этих интервалов. У показателя с отклонениями есть специальная пиктограмма с восклицательным знаком в желтом треугольнике, которая помогает сразу его заметить. С этого экрана можно перейти в детали конкретного показателя, а также отредактировать текущий анализ.

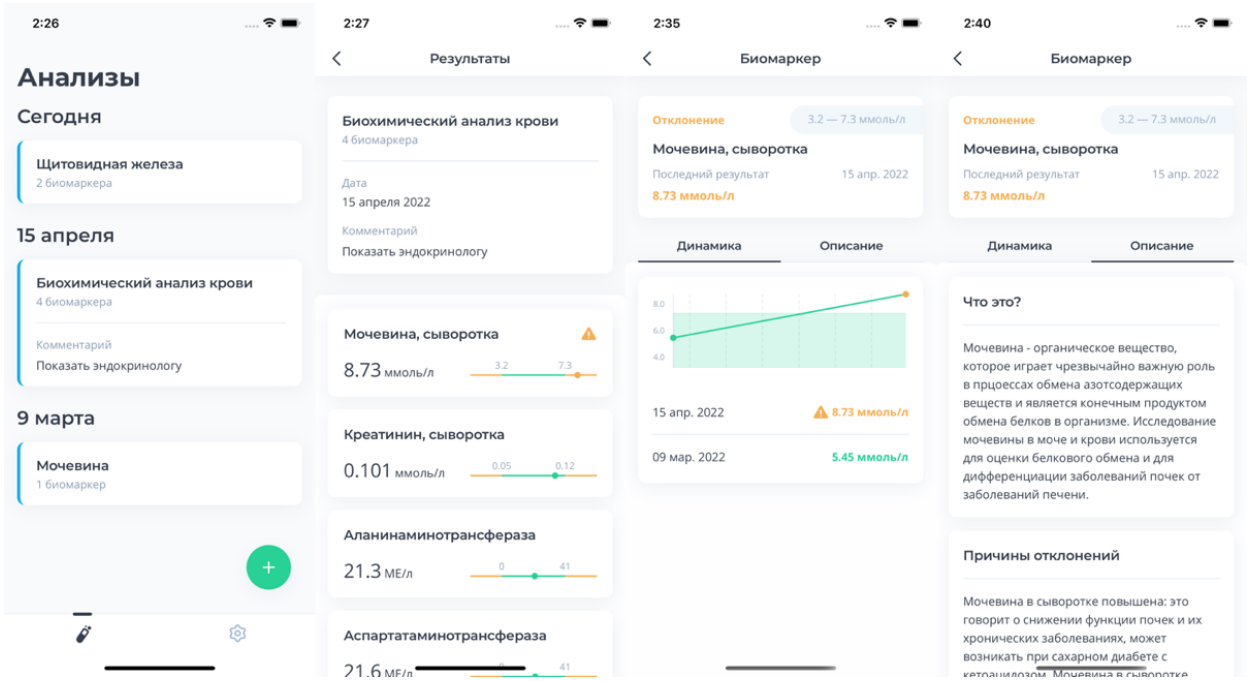

Рисунок 20 - Экраны просмотра анализов в мобильном приложении

В деталях показателя анализа отображены референсные интервалы, показано значение последнего анализа и соответствует ли оно норме. При отклонении от нормы цвет меняется на желтый, если все в норме, то на зеленый. Ниже представлен график со всеми значениями из предыдущих анализов, а также приведен список с датами этих анализов. На основе этого отслеживается динамика показателей. На вкладке «Описание» можно посмотреть более подробное описание показателя, а также ознакомиться с причинами отклонения. Это поможет расшифровать причину отклонения.

Так же с главного экрана можно добавить новый анализ (рисунок 21).

При нажатии на зеленую кнопку со знаком «+»откроется экран добавления. Он состоит из 4-х шагов:

– Добавление показателей;
– Заполнение информации об анализе;
– Подтверждение результатов;
– Уведомление об успешном добавлении.

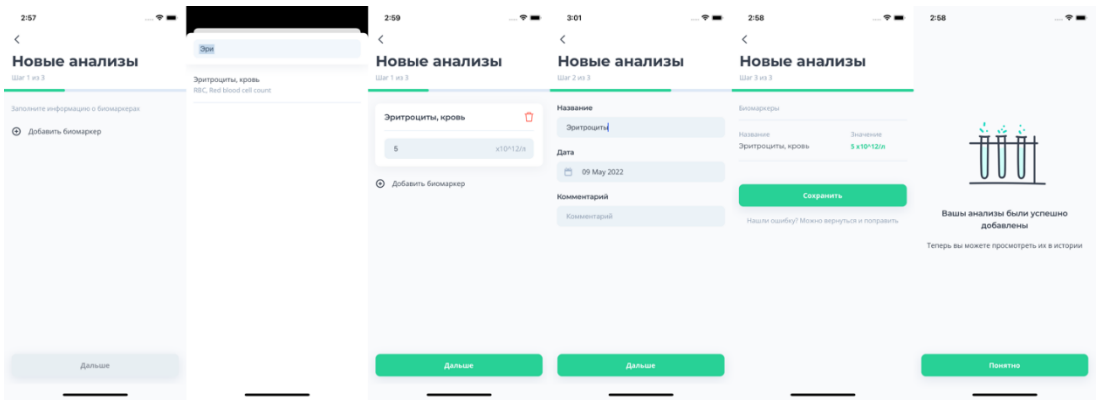

Рисунок 21 - Экраны добавления анализа в мобильном приложении

На первом шаге осуществляется добавление результатов анализа для конкретных показателей. При нажатии на кнопку «Добавить» открывается модальное окно, в котором приводится список со всеми доступными показателями. При необходимости можно выполнить поиск. После выбора показателя вводится результат в соответствующее поле ввода. Рядом с полем ввода указана единица измерения. На следующем шаге указывается название анализа, которое будет отображаться в списке. Так же указывается дата анализа. По желанию добавляется комментарий к анализу. На последнем шаге добавления отображаются все введенные показатели и их результаты. Это необходимо, чтобы перепроверить результаты перед окончательным добавлением. После успешного добавления анализа показывается соответствующий экран.

3.7 Тестирование информационной системы расшифровки анализов крови

3.7.1 Тестирование серверного приложения

Для тестирования серверного приложения выбран способ ручного тестирования. Для этого необходимо запустить приложение в разных браузерах и выполнить все тест-кейсы.

Для тестирования были выбраны следующие популярные браузеры на момент тестирования:

– Google Chrome (101.0.4951.54);
– Apple Safari (15.2);
– Mozilla Firefox (101.0).

Тест-кейсы для тестирования системы управления контентом представлены в таблице 3.

Таблица 3 - Тест-кейсы для тестирования системы управления контентом

Описание	Результат
Аутентификация врача	Пройден
Добавление анализов	Пройден
Редактирование анализов	Пройден
Добавление референсных значений	Пройден
Редактирование референсных значений	Пройден
Добавление информации об отклонениях	Пройден
Редактирование информации об отклонениях	Пройден
Добавление единиц измерения показателей	Пройден
Редактирование единиц измерения показателей	Пройден

Все тест-кейсы были успешно пройдены на каждом устройстве.

3.7.2Тестирование клиентского приложения

По аналогии с серверным приложением для тестирования мобильного приложения необходимо провести ручное тестирование. Для этого приложение нужно запустить на разных версиях платформ iOSи Android. Для тестирования использовались симулятор iOSи эмулятор Androidустройств преимущественно последних версий.

Для iOSбыли выбраны следующие устройства:

– iPhone 11 (iOS 15.2);
– iPhone 12 Pro (iOS 15.3);
– iPhone 13 Pro Max (iOS 15.4).

Для Androidиспользовались следующие устройства:

– Pixel 5 (Android 11.0);

– Pixel XL (Android 12.0).

На основе функциональных требований к мобильному приложению были составлены тест-кейсы, представленные в таблице 4.

Таблица 4- Тест-кейсы для тестирования мобильного приложения

Описание	Результат
Аутентификация пациента	Пройден
Регистрация пациента	Пройден
Добавление анализов	Пройден
Редактирование анализов	Пройден
Просмотр списка анализов	Пройден
Просмотр деталей анализов	Пройден
Просмотр деталей конкретного показателя анализа	Пройден

Все указанные тест-кейсы в выбранных браузерах были успешно пройдены.

На основе этого делается вывод, что информационная система расшифровки анализов крови работает корректно, согласно всем требованиям.

Вывод по третьей главе

В результате данной главы была разработана информационная система расшифровки анализов крови. Была выбрана трехзвенная архитектура ИС, выбраны библиотеки для реализации, спроектированы структуры и реализованы серверное приложение и мобильное приложение. После этого было проведено ручное тестирование, которое не выявило никаких отклонений от требований.

Заключение

В процессе выполнения бакалаврской работы была спроектирована и разработана информационная система расшифровки анализов крови. Было реализовано серверное приложение при помощи фреймворка Strapiи мобильное приложение при помощи фреймворка ReactNative.

На начальном этапе была сформулирована актуальность темы, определены объект, предмет, цели и задачи.

В первой главе был проведен анализ предметной области расшифровки анализов крови и проведено моделирование бизнес-процесса с помощью диаграммы «КАКЕСТЬ». Так же был проведен анализ схожих решений. На основе этого были составлены требования к информационной системе. Затем была разработана диаграмма бизнес-процесса расшифровки анализов крови «КАК ДОЛЖНО БЫТЬ».

Во второй главе была выбрана технология логического моделирования информационной системы расшифровки анализов крови. После составлена таблица прецедентов и построены диаграммы вариантов использования, последовательности и классов. После этого было произведено логическое моделирование базы данных.

В третьей главе была выбрана трехзвенная архитектура ИС, выбраны библиотеки для реализации, спроектированы структуры и реализованы серверное приложение и мобильное приложение. На основе этого была разработана информационная система расшифровки анализов крови. После было проведено ручное тестирование приложений.

Были решены следующие задачи:

- проведен анализ области автоматизации;
- составлены модели бизнес-процессов;
- проанализированы существующие решения;
- спроектирована база данных;

– спроектирована архитектура серверной части и мобильного приложения;
– проведено сравнение и выбор технологий и библиотек для реализации;
– реализовано серверное приложение и мобильное приложение для информационной системы расшифровки анализов крови;
– проведено ручное тестирование.

По результатам работы цель бакалаврской работы была достигнута.

Разработанная информационная система предоставляет возможность для хранения и расшифровки анализов крови. Эта информационная система поможет людям быть более осведомленными о результатах анализов и своем здоровье.

Готовая информационная система расшифровки анализов крови в дальнейшем может быть легко модернизирована.

Список используемой литературы

1. ГОСТ Р 53022.1-2008 Технологии лабораторные клинические. Требования к качеству клинических лабораторных исследований. Часть 1. Правила менеджмента качества клинических лабораторных исследований. Введен 2010-01-01.- М.: Стандартинформ, 2009. – 10с.

2. ГОСТ Р ИСО 15189-2015 Лаборатории медицинские. Частные требования к качеству и компетентности. Взамен ГОСТ Р ИСО 15189-2009. Введен 2016-06-01.- М.: Стандартинформ, 2015. – 5с.

3. Интерпретация результатов лабораторных исследований. Информационное письмо / О.Н. Сизикова, Л.Р. Колесниченко – Чита: ГУЗ Краевая клиническая больница, 2017. – С. 2-5.

4. Реинжиниринг бизнес-процессов. Учебное пособие / А.О. Блинов, О.С. Рудакова, В.Я. Захаров, И.В. Захаров – М.: ЮНИТИ-ДАНА, 2015. — 343 с.

5. Трёхзвенная архитектура [Электронный ресурс]: Описание архитектуры. URL: https://studme.org/282494/informatika/tryohzvennaya_arhitektura (Дата обращения 09.04.2022).

6. Apollo Docs Home [Электронный ресурс]: Сайт с документацией к библиотеке Apollo. URL: https://www.apollographql.com/docs/ (Дата обращения 13.04.2022).

7. Biogenom: Медицинское мобильное приложение [Электронный ресурс]: Сайт приложения Biogenom. URL: https://biogenom.ru/ (Дата обращения 18.03.2022).

8. Bonnie Eisenman, Learning React Native: Building Native Mobile Apps with JavaScript. – 2nd Edition, O'Reilly Media, Inc., 2017.

9. Boris Cherny, Programming TypeScript: Making Your JavaScript Applications Scale. – 1st Edition, O'Reilly Media, Inc., 2019.

10. David Flanagan, JavaScript: The Definitive Guide. – 7th Edition, O'Reilly Media, Inc., 2020.

11. Eve Porcello, Alex Banks: Learning GraphQL. – 1st Edition, O'Reilly Media, Inc., 2018.

12. GraphQL is the better REST [Электронныйресурс]: Сравнение GraphQL и REST API. URL: https://www.howtographql.com/basics/1-graphql-is-the-better-rest/ (Дата обращения 04.04.2022).

13. IntroductionReactNative [Электронный ресурс]: Сайт с документацией к библиотеке ReactNative. URL: https://reactnative.dev/docs/getting-started (Дата обращения 24.04.2022).

14. Introductionto GraphQL Code Generator [Электронный ресурс]: Сайт с документацией к библиотеке @graphql-codegen. URL: https://www.graphql-code-generator.com/docs/getting-started (Дата обращения 26.04.2022).

15. Introductionto GraphQL [Электронный ресурс]: Сайт с документацией к GraphQL. URL: https://graphql.org/learn/ (Дата обращения 02.04.2022)

16. Khalid Elshafie, Mozafar Haider: Designing Web APIs with Strapi. – 1st Edition, O'Reilly Media, Inc., 2022.

17. Ornament: Система управления здоровьем [Электронный ресурс]: Сайт приложения Ornament. URL: https://ornament.health/ru (Дата обращения 19.03.2022).

18. PostgreSQLDocumentation [Электронный ресурс]: Сайт с документацией к PostgreSQL. URL: https://www.postgresql.org/docs/current/ (Дата обращения 15.04.2022).

19. UML для бизнес-моделирования: зачем нужны диаграммы процессов [Электронный ресурс]: Сайт с UML диаграммами. URL: https://evergreens.com.ua/ru/articles/uml-diagrams.html (Дата обращения 15.03.2022).

20. Welcome to the Strapi v4 developer documentation [Электронныйресурс]: СайтсдокументациейкStrapi. URL:

https://docs.strapi.io/developer-docs/latest/getting-started/introduction.html (Дата обращения 20.04.2022)

Приложение А

Листинг кода физической модели БД на языкеDBML

```
Table doctors {
  id int [pk, increment] // auto-increment
firstname varchar
lastname varchar
  birthdate timestamp
  gender varchar
  email varchar
  password varchar
created_at timestamp
updated_at timestamp
}

Table patients {
  id int [pk, increment] // auto-increment
firstname varchar
lastname varchar
  username varchar
  birthdate timestamp
  gender varchar
  email varchar
  password varchar
created_at timestamp
updated_at timestamp
}

Table categories {
  id int [pk]
  name varchar
created_at timestamp
updated_at timestamp
}
Table biomarkers {
  id int [pk]
category_id int [ref: > categories.id]
  name varchar
```

```
descriptiontext
deviation text
  synonyms varchar
created_at timestamp
updated_at timestamp
}

Table units {
  id int [pk]
  name varchar
  synonyms varchar
  type varchar
created_at timestamp
updated_at timestamp
 }

Table references {
  id int [pk]
unit_id int [ref: > units.id]
biomarker_id int [ref: - biomarkers.id]
created_at timestamp
updated_at timestamp
}

Table reference_entries {
  id int [pk]
reference_id int [ref: > references.id]
  gender varchar
value_from double
value_to double
time_period_from int
time_period_to int
created_at timestamp
updated_at timestamp
}
Table submissions {
  id int [pk]
```

```
patient_id int [ref: > patients.id]
test_datetimestamp
name varchar
  comment varchar
created_at timestamp
updated_at timestamp
}

Table submission_entries {
  id int [pk]
submission_id int [ref: > submissions.id]
biomarker_id int [ref: > biomarkers.id]
  value int
created_at timestamp
updated_at timestamp
}
```

Приложение Б

Подключение Strapi к базе данных

```
module.exports = ({ env }) => ({
  connection: {
    client: "postgres",
    connection: {
      host: env("DATABASE_HOST", "127.0.0.1"),
      port: env.int("DATABASE_PORT", 5432),
      database: env("DATABASE_NAME", "aid"),
      user: env("DATABASE_USERNAME", "kondratjev"),
      password: env("DATABASE_PASSWORD", "kondratjev"),
ssl: env.bool("DATABASE_SSL", false),
},
  },
});
```

Приложение В

Листинг файла с описанием модели данных

```
{
  "kind": "collectionType",
  "collectionName": "submissions",
  "info": {
    "singularName": "submission",
    "pluralName": "submissions",
    "displayName": "Submission",
    "description": ""
  },
  "options": {
    "draftAndPublish": false
  },
  "pluginOptions": {},
  "attributes": {
    "user": {
      "type": "relation",
      "relation": "oneToOne",
      "target": "admin::user"
    },
    "testDate": {
      "type": "date",
      "required": true
    },
    "entries": {
      "type": "component",
      "repeatable": true,
      "component": "submission.submission",
"required": true
},
    "comment": {
      "type": "string",
"required": false
    }
}
}
```

Приложение Г

Подключение библиотеки @graphql-codegen/cli

```
overwrite: true
errorsOnly: true
schema: "http://localhost:4000/graphql"
documents: "src/**/*.graphql.ts"
generates:
src/generated/graphql.tsx:
   plugins:
     - "typescript"
     - "typescript-operations"
     - "typescript-react-apollo"
```