

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования

«Кубанский государственный технологический университет»
(ФГБОУ ВО «КубГТУ»)


Институт компьютерных систем и информационной безопасности
Кафедра информационных систем и программирования


ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

по направлению: 09.04.04 Программная инженерия
(код и наименование направления)


на тему: «Мобильное приложение для функциональной диагностики»
(наименование темы)

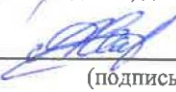
ИКСИБ.КИСП.09.04.04.001.ПП
(обозначение документа)

Автор  06.06.2022 / О. Нечнае /
(подпись, дата, расшифровка подписи)

Руководитель  07.06.2022 / канд. техн. наук, доц. В.А. Мурлина /
(подпись, дата, расшифровка подписи)

Консультанты:

Раздел информационной
безопасности проекта  08.06.22 / канд. техн. наук, доц. В.А. Шарай /
(подпись, дата, расшифровка подписи)

Нормоконтролер  15.06.2022 / ст. преп. А.А. Ковтун /
(подпись, дата, расшифровка подписи)

Выпускная квалификационная работа
допущена к защите

17.06.2022
(дата)

Заведующая кафедрой  / канд. техн. наук, доц. М.В. Янаева /
(подпись)

Краснодар
2022

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования

«Кубанский государственный технологический университет»
(ФГБОУ ВО «КубГТУ»)

Институт компьютерных систем и информационной безопасности
Кафедра информационных систем и программирования

УТВЕРЖДАЮ
Заведующая кафедрой ИСП
к.т.н, доц. М.В. Янаева
«18» апреля 2022 г.

ЗАДАНИЕ НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

по направлению 09.04.04 Программная инженерия
(шифр и наименование)

студенту Нечначе Оуссама
(фамилия, имя, отчество)

Тема выпускной квалификационной работы: «Мобильное приложение для функциональной диагностики»

Утверждена приказом ректора университета от 15.04.2022 г. № 634-Ст

Руководитель канд. техн. наук, доц. В.А. Мурлина
(должность, фамилия, инициалы)

Консультанты:

Раздел информационной
безопасности проекта канд. техн. наук, доц. В.А. Шарай
(должность, фамилия, инициалы)

Нормоконтролер ст. преп. А.А. Ковтун
(должность, фамилия, инициалы)

Срок сдачи выпускной квалификационной работы на кафедру 17.06.2022 г.

Содержание выпускной квалификационной работы

Введение

- 1 Нормативные ссылки
- 2 Термины и определения
- 3 Сокращения
- 4 Описание объекта статистического анализа
- 5 Разработка структурной схемы приложения
- 6 Реализация приложения
- 7 Информационная безопасность

Заключение

Список использованных источников

Приложение А – Листинг программы

Общее количество листов ПЗ 72

Объем иллюстративной части

1. Описание объекта статистического анализа
2. Разработка структурной схемы приложения
3. Реализация приложения
4. Информационная безопасность
5. Выводы из проделанной работы


Общее количество слайдов иллюстративной части 11

Список основной и рекомендуемой литературы

1. Методические указания по выполнению выпускной квалификационной работы для студентов всех форм обучения и МИПС направлений 09.03.03 Прикладная информатика, 09.03.04 Программная инженерия / Сост.: Л.А. Видовский, М.В. Янаева; Кубан. гос. технол. ун-т. Каф. информационных систем и программирования. – Краснодар: Изд. КубГТУ, 2016. – 44 с.

Календарный план выполнения выпускной квалификационной работы

Месяц	Числа месяца																														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Апрель																		Описание требований к ИС и постановка задачи ВКР													
Май	Технологий для реализации приложения												Выбор программного средства и программная реализация										Тестирование приложения								
Июнь	Информационная безопасность								Заключение , презентация , рецензирование									Защита ВКР													

Студент  18.04.22 О. Нечнае
(подпись, дата)

Руководитель  18.04.22 канд. техн. наук, доц. В.А. Мурлина
(подпись, дата)

Реферат

Выпускная квалификационная работа содержит 72 страницы, 27 рисунков, 3 таблиц, 9 источников, 1 приложение.

БЫСТРАЯ ПОМОЩЬ, БОЛЕЗНИ, МОНИТОРИНГ, МОБИЛЬНАЯ РАЗРАБОТКА, МАШИННОЕ ОБУЧЕНИЕ, ANDROID STUDIO, JAVA, API, XML, DATA STRUCTURE, ЛИНЕЙНАЯ РЕГРЕССИЯ.

Целью выпускной квалификационной работы является разработка мобильной приложения на тему «Мобильное приложение для функциональной диагностики».

В результате выполнения выпускной квалификационной работы относятся мобильное приложение для функциональной диагностики оказывает помощь пациентам, диагностируя его заболевание и собирая его данные для получения правильного результата. также поможем пациенту добраться до врача для консультации и предложим больницы и лекарства для типа его заболевания, а также ближайшие к нему аптеки

Программное обеспечение создано на языке Java с помощью SDK набор инструментов для разработки программного обеспечения в одном устанавливаемом пакете.

					ИКСИБ.КИСП.09.04.04.001.ПП			
Изм.	Лист	№ докум.	Подпись	Дата	Пояснительная записка выпускной квалификационной работы	Лит.	Лист	Листов
Разраб.		Нечначе О.		06.06				
Провер.		Мурлина В.А.		07.06			6	72
Консульт.		Шарай В.А.		08.06		КуДГТУ		
Н. Конт		Ковтун А.А.		15.06.22				
Утверд.		Янаева М.В.		17.06.22				

Содержание

Введение.....	8
1 Нормативные ссылки.....	9
2 Термины и определения	10
3 Сокращения.....	12
4 Описание объекта статистического анализа	13
4.1 Общее описание предметной области	13
4.2 Выбор технологии.....	14
4.2.1 Выбор языка программирование.....	14
4.2.2 Выбор библиотеки.....	19
4.2.3 Выбор алгоритма.....	20
4.2 Аналоги разрабатываемого приложения	22
5 Разработка структурной схемы приложения.....	28
5.1 Диаграмма сущность-связь	28
5.2 Диаграмма классов.....	30
5.3 Диаграмма вариантов использования	31
5.4 Разработка IDEF0 и IDEF3 диаграмм	34
5.5 Диаграмма активности.....	38
5.6 Диаграмма последовательности	39
6 Реализация приложения	41
6.1 Предварительные установки.....	41
6.2 Разработка функциональной модели	41
6.3 Разработка технической и справочной информации.....	43
7 Информационная безопасность	50
7.1 Построение диаграммы потоков данных.....	50
7.2 Выявление перечня угроз	53
7.3 Построение деревьев опасностей	55
7.4 Анализ рисков опасности	61
7.5 Меры по предотвращению опасностей.....	62

Закключение	63
Список использованных источников	64
Приложение А Листинг программы.....	65

Введение

Здоровье одна из наивысших ценностей человека. Вполне очевидно, что хорошее здоровье является основным условием для нормальной реализации человеком биологического и социального функционала. Это фундамент самореализации личности. Каждый из нас понимает и осознает, что нездоровье или болезнь – это жизнь, ограниченная в своей свободе.

Известно, что если у человека возникают определенные болезненные симптомы, то ему необходимо обратиться к врачу. При посещении врача человек описывает свои симптомы, а врач проводит необходимую функциональную диагностику человека.

Однако, обычно в таких случаях люди обращаются за помощью не к врачу, а к сети интернет, где находят определенного рода статьи и комментарии, связанные с их симптоматикой. Зачастую все мнения и рекомендации являются ошибочными и сильно разнятся на различных информационных ресурсах, поэтому это программное обеспечение и использование машинного обучения могут ускорить анализ данных за секунды без проблем и в данных, а также с точностью. Анализ Эта программа предоставляет множество услуг.

В ходе научно-исследовательской ВКР были изучены информационные технологии, техническое оборудование, программные средства и базы данных, которыми располагает КубГТУ. Разработана программа на языке Java с помощью SDK набор инструментов для разработки программного обеспечения в одном устанавливаемом пакете. Они облегчают создание приложений, имея компилятор, отладчик и иногда программную среду. В основном они зависят от комбинации аппаратной платформы компьютера и операционной системы.

1 Нормативные ссылки

В настоящей ВКР использованы ссылки на следующие руководящие документы:

1. ГОСТ 19.404-79 Государственный стандарт. Единая система программной документации пояснительная записка. Требования к содержанию и оформлению.

3. ГОСТ 19.701-90. (ИСО 5807-85). ЕСПД. Государственный стандарт. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения.

4. ГОСТ 19.781-90. Государственный стандарт. Обеспечение систем обработки информации программное. Термины и определения.

5. ГОСТ Р 51904-2002 Группа П85. Государственный стандарт Российской федерации. Программное обеспечение встроенных систем. Общие требования к разработке и документированию.

6. ГОСТР 54593–2011 Информационные технологии. Государственный стандарт российской федерации. Свободное программное обеспечение.

7. ГОСТ 3.1105-2011 Государственный стандарт Российской федерации. Единая система технологической документации. Формы и правила оформления документов общего назначения.

8. ГОСТ Р ИСО/МЭК 12119-2000 Информационная технология. Пакеты программ. Требования к качеству и тестирование.

2 Термины и определения

В настоящей выпускной квалификационной работе применяются термины с соответствующими определениями и сокращениями, установленные нормативным документом ГОСТ Р 1.12-2004. Стандартизация в Российской Федерации. Термины и определения:

1 UML – язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования бизнес-процессов, системного проектирования и отображения организационных структур.

2 Информационная система (ИС) – это система, реализующая информационную модель предметной области, чаще всего – какой-либо области человеческой деятельности. ИС должна обеспечивать: получение (ввод или сбор), хранение, поиск, передачу и обработку (преобразование) информации.

3 Machine Learning – машинное обучение, класс методов искусственного интеллекта, характерной чертой которых является не прямое решение задачи, а обучение за счёт применения решений множества сходных задач.

4 Библиотека – это набор готовых функций, классов и объектов для решения каких-то задач.

5 Проект – комплекс взаимосвязанных мероприятий, предназначенных для достижения, в течение заданного периода времени и при установленном бюджете, поставленных задач с четко определенными целями.

6 Сервер – выделенный или специализированный компьютер для выполнения сервисного программного обеспечения (в том числе серверов тех или иных задач).

7 Linear regression – линейная регрессия, используемая в статистике регрессионная модель зависимости одной (объясняемой, зависимой).

8 Gradle – система для автоматизации сборки приложений и сбора статистики об использовании программных библиотек, применяющая языки Groovy, Java, JavaScript, Kotlin и т. д., а также решения из фреймворков Apache Ant и Apache Maven.

9 Аутентификация – Проверка принадлежности субъекту доступа предъявленного им идентификатора, подтверждение подлинности.

10 Безопасность информации – состояние защищенности информации, характеризуемое способностью персонала, технических средств и информационных технологий обеспечивать конфиденциальность.

11 ER диаграмма – диаграмма «сущность-связь». Представляет связи между объектами с их атрибутами в наглядном виде.

3 Сокращения

В данной выпускной квалификационной работе использованы следующие сокращения:

- 1 **ИС** – информационная система;
- 2 **СУБД** – система управления базами данных;
- 3 **SDK** – комплект для разработки программного обеспечения;
- 4 **ИИ** – искусственный интеллект;
- 5 **ПП** – программный продукт;
- 6 **ОС** – операционная система.

4 Описание объекта статистического анализа

4.1 Общее описание предметной области

За последние несколько лет человечество успел реализовать много различных приложений и везде внедряются технологии. Соответственно сейчас уже существует ПП во всех областях науки и повседневной жизни, начиная от сельскохозяйственных до инновационных промышленности, от малого до крупного.

На сегодняшний день существуют огромное количество ПП в самых разных областях, но тем не менее, прогресс не стоит на месте, полезные программы продолжают совершенствоваться.

С каждым годом в России и во всем мире увеличиваются темпы роста заболеваний, соответственно это представляет большую опасность для здоровья людей и, следовательно, это оказывает высокое давление на систему здравоохранения.

Для того чтобы решить выше перечисленные задачи данная программа предлагает:

- определение заболеваний;
- определение состояния здоровья человека;
- консультирование;
- запись всех заболеваний и история учета;
- нахождение ближайших и открытых больниц, аптек и т.д.;
- информация о врачах, больницах и аптеках;
- опыт работы и сделанные операции (сертификаты) врача;
- информация о болезнях и их симптомах.

Одной из сложностей создания приложения является отсутствие информации о больницах и аптеках и времени их работы, а также отсутствие базы данных, содержащей данные о количестве заболеваний.

В данной научно-исследовательской работе рассматриваются способы реализовать программу с искусственным интеллектом и тщательно

исследовать состояние пациентов. Соответственно, для разработки этой программы необходимо изучить новые алгоритмы и технологии, выбрать оптимальный вариант и предоставить методы их решения.

Реализуемая программа в дальнейшем может расширять свои возможности, но для начального уровня этого уже достаточно для решения многих проблем.

4.2 Выбор технологии

4.2.1 Выбор языка программирование

Приложения для Android разрабатываются с использованием языка программирование Java. На данный момент – это оптимальный вариант для нативных приложений. java очень популярный язык программирования, разработанный Sun Microsystems (сейчас принадлежит Oracle). Разработанный намного позже C и C++, Java включает в себя многие функций этих языков, но при этом устраняя некоторые из их недостатков.

Некоторые из важных основных функций Java:

- легкость освоения;
- безопасность, отсутствие привязки к платформе;
- виртуальные машины;
- объектно-ориентированность;
- независимость платформа;
- высокая производительность;
- динамичность.

Android в значительной степени опирается на эти основы Java. Android SDK включает в себя множество стандартных библиотек Java (библиотеки структур данных, математические библиотеки, графические библиотеки, сетевые библиотеки и все остальное, что может понадобиться), а также специальные библиотеки Android, которые помогут вам разрабатывать потрясающие приложения для Android.

Java Vs Kotlin – Есть много различий и сходств между языком программирования Kotlin и Java, сравнение в таблице 4.1.

Таблица 4.1 – Разница между Kotlin и Java

	Kotlin	Java
Возраст языка	Kotlin появился в 2011 году, и молодость языка в этом случае не играет ему на руку. Чтобы найти ответ на какой-то вопрос или решение задачи, придётся потратить немало времени. Даже документация здесь не сильно поможет. В большинстве случаев она сводится к тому, что задачи решаются похоже с Java, но с небольшими изменениями	Язык появился в 1995 году. За это время вокруг него сформировалось большое комьюнити, и это позволяет без проблем найти ответ на вопрос, даже очень специфичный. Вы можете почитать документацию, посмотреть видеоролик или обратиться к другим разработчикам – с вероятностью 99% ответ будет найден
Количество кода	Чтобы написать на Kotlin такую же программу, как на Java, у вас уйдёт меньше времени и строк кода. Это очень удобно, когда требуется создать проект в короткие сроки	Особенности синтаксиса Java предполагают, что код будет более громоздким, чем в Kotlin. Вдобавок язык крайне консервативен.

Проверяемые исключения	<p>В языке не предусмотрены условия для проверяемых исключений, а значит нет необходимости объявлять какие-либо исключения.</p> <p>Код сокращается – безопасность повышается</p>	<p>Проверяемые исключения – головная боль Java программистов.</p> <p>Компилятор заставляет каждый привлекающий внимание метод обозначать как исключение. Разработчик вынужден обрабатывать его, а это сказывается на скорости разработки</p>
Безопасность	<p>В Kotlin по умолчанию встроена null-безопасность поэтому если программист решит присвоить чему-то значение null – во время компиляции просто произойдёт сбой. Для разработчиков null-безопасность языка позволяет не писать дополнительный код, что является очередным преимуществом языка перед Java</p>	<p>Ещё одна проблема Java разработчиков связана с так называемой «нулевой безопасностью» или исключением <i>NullPointerException</i>. В Android используется <i>null</i> для представления отсутствия значений, но эта же ситуация может попросту уничтожить программу</p>

<p>Среда разработки</p>	<p>Android Studio базируется на программе IntelliJ Idea, которая создана и поддерживается той же компанией, что и Kotlin – JetBrains. Поэтому Kotlin-разработчики получают актуальную IDE, которая покрывает все их запросы</p>	<p>При разработке приложений под Android в Java используется специализированная среда разработки IDE – Android Studio. Она заточена под работу с Java, поэтому вы можете начинать писать строку, а IDE сама её закончит</p>
<p>Цели разработки</p>	<p>Если вы хотите создать приложение с долгим сроком жизни, то оптимальным вариантом будет выбор Kotlin. Особенно если учитывать простоту разработки на нём и множество фишек, которых нет в Java</p>	<p>Большинство приложения для Android написаны на Java, и маловероятно, что кто-то решит переписать их на Kotlin. Также, когда дело касается инновационных решений, VR/AR-приложений или роботомедицины</p>

Перспективы	<p>Kotlin – молодой и бунтарский язык, который идёт вразрез с консервативными устоями предков. Он похож на подростка. Многие современные компании переходят на этот язык, но всё ещё неизвестно, что станет с ним завтра. Возможно, скоро он обгонит по популярности язык Java, а возможно – повторит судьбу мёртвого языка Perl</p>	<p>С развитием технологий универсальный Java точно не потеряет свою популярность. Он покоряет своей кроссплатформенностью и возможностью программировать на нём ПО для чего угодно – даже для будильника. Умирать в ближайшее время он не собирается точно, об этом говорит хотя бы его использование в космических разработках</p>
-------------	--	---

Java ООП – Процедурное программирование написание процедур или методов, выполняющих операции с данными, в то время как объектно-ориентированное программирование – это создание объектов, содержащих как данные, так и методы.

Объектно-ориентированное программирование имеет несколько преимуществ перед процедурным программированием:

- ООП быстрее и проще в исполнении;
- ООП обеспечивает четкую структуру программ;
- ООП помогает сохранить код Java СУХИМ «Не повторяйтесь» и упрощает поддержку, изменение и отладку кода;
- ООП позволяет создавать полностью повторно используемые

приложения с меньшим количеством кода и более коротким временем разработки.

XML – означает расширяемый язык разметки. Он был создан как стандартный способ кодирования данных в интернет-приложениях. XML сам по себе хорошо читается как человеком, так и машиной. Кроме того, он масштабируется и прост в разработке. В Android мы используем XML для разработки наших макетов, потому что XML – легкий язык, поэтому он не утяжеляет наш макет. пользовательский интерфейс нашего приложения – это все, что имеет значение для пользователя, это то, что привлекает пользователей и что отталкивает их, и поэтому настоятельно рекомендуется полировать наш пользовательский интерфейс, но никогда не игнорировать ядро нашего приложения.

4.2.2 Выбор библиотеки

Библиотеки платформы включают в себя различные основные библиотеки C/C++ и библиотеки на основе Java, такие как Media, Graphics, Surface Manager, OpenGL и т. д., чтобы обеспечить поддержку разработки для Android:

- Media library обеспечивает поддержку воспроизведения и записи аудио и видео форматов.
- Surface manager отвечает за управление доступом к подсистеме отображения.
- SQLite обеспечивает поддержку базы данных, а FreeType – поддержку шрифтов.
- Web-Kit этот движок веб-браузера с открытым исходным кодом предоставляет все функции для отображения веб-контента и упрощения загрузки страниц.
- SSL (Secure Sockets Layer) это технология безопасности, позволяющая установить зашифрованную связь между веб-сервером и веб-

браузером.

– Android SDK разработанный Google для платформы Android. С помощью Android SDK мы можем легко создавать приложения для Android. Он представляет собой набор библиотек и инструментов разработки программного обеспечения, которые необходимы для разработки приложений для Android. Android SDK состоит из нескольких инструментов, которые очень важны для разработки приложений для Android. Эти инструменты обеспечивают плавный переход от разработки к отладке. Android SDK совместим со всеми операционными системами, такими как Windows, Linux, macOS и т. д.

API (Application Programming Interface) – это набор инструкций и стандартов программирования для доступа к веб-инструменту или базе данных. Компания-разработчик программного обеспечения публикует свой API для широкой публики, чтобы другие разработчики программного обеспечения могли разрабатывать продукты, основанные на ее сервисе. API обычно упаковывается в SDK.

4.2.3 Выбор алгоритма

Supervised machine learning – В supervised learning, алгоритм машинного обучения обучен правильно отвечать на вопросы, связанные с векторами признаков. Для обучения алгоритма машине подается набор векторов признаков и соответствующая метка. Метки обычно предоставляются комментатором-человеком и представляют собой правильный «ответ» на заданный вопрос. Алгоритм обучения анализирует векторы признаков и их правильные метки, чтобы найти внутренние структуры и отношения между ними. Таким образом, машина учится правильно реагировать на запросы.

исходные предварительно обработанные наборы данных, содержащие известные переменные и цели, делятся на обучающие данные и тестовые

данные. (Вверху) На этапе обучения данные обучения используются для обучения алгоритма обучения в попытке разработать точную прогностическую модель. (В центре) Чтобы проверить модель, тестовые данные затем применяются к модели и оценивается прогностическая точность. (Ниже) После проверки новые данные вводятся в модель в попытке сделать новые прогнозы. (рисунок 4.1).

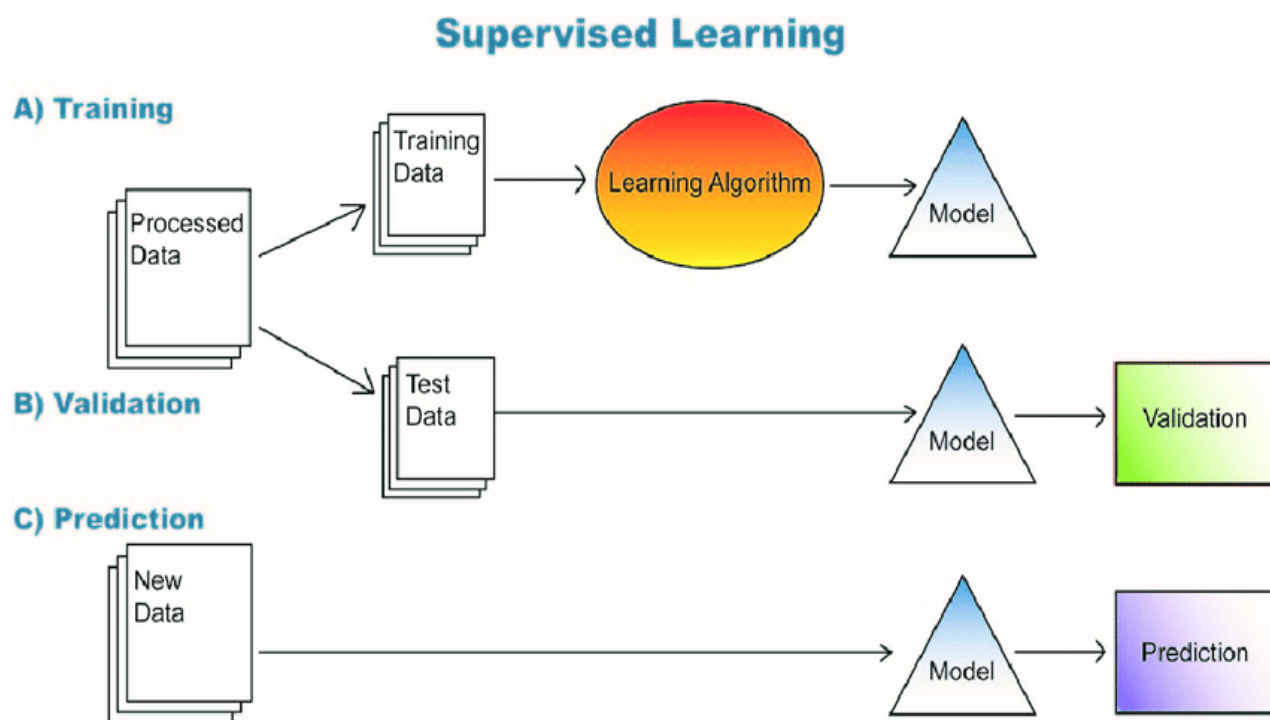


Рисунок 4.1 – Модель Supervised machine learning

Цель машинного обучения состоит в том, чтобы определить целевую функцию, которая будет работать максимально точно для неизвестных, невидимых экземпляров данных. В машинном обучении целевую функцию ($h\theta$) иногда называют моделью. Эта модель является результатом процесса обучения.

Linear regression – в функции линейной регрессии тета-параметры и параметры признаков нумеруются по номеру подписки. Номер подписки указывает положение тета-параметров (θ) и параметров признаков (x) в векторе. Обратите внимание, что функция x_0 представляет собой постоянный термин смещения, установленный со значением 1 для вычислительных

целей. В результате индекс специфической для домена функции, такой как размер дома, будет начинаться с x_1 . Например, если x_1 установлено для первого значения вектора характеристик дома, размер дома, то x_2 с использованием уравнения (4.1).

$$h_0 = \theta_0 * 1 + \theta_1 * x_1 + \dots + \theta_n * X_n \quad (4.1)$$

Чтобы научить машину думать, первым шагом будет выбор алгоритма обучения, который вы будете использовать. Линейная регрессия – один из самых простых и популярных алгоритмов обучения с учителем. Этот алгоритм предполагает, что связь между входными объектами и выходной меткой является линейной. Приведенная ниже общая функция линейной регрессии возвращает прогнозируемое значение путем суммирования каждого элемента вектора признаков, умноженного на тета-параметр (θ). Тета-параметры используются в процессе обучения для адаптации или «настройки» функции регрессии на основе данных обучения.

4.2 Аналоги разрабатываемого приложения

Прежде чем приступить к разработке собственной программы, нам следует внимательно прочитать и изучить примеры реализации на реальных примерах, желательно на примерах успешных и популярных приложений, где у вас больше шансов сделать все правильно, следуя всем вышеперечисленным принципам разработки.

Symptom Checker – уникальный проект, сочетающий медицину с компьютерным искусственным интеллектом, разработанным профессиональными немецкими врачами (рисунок 4.2).

В контексте смоделированного разговора между врачом и пациентом вы получаете вопросы, на которые можете ответить индивидуально. Цель состоит в том, чтобы найти предполагаемый диагноз на основе ваших

ответов о ваших симптомах и предоставить вам индивидуальную информацию, которая вам понадобится для возможной встречи с вашим настоящим врачом.

Средство проверки симптомов поможет вам понять, как врач анализирует ваши симптомы. Это интерактивное приложение предназначено для людей, не имеющих медицинского образования.

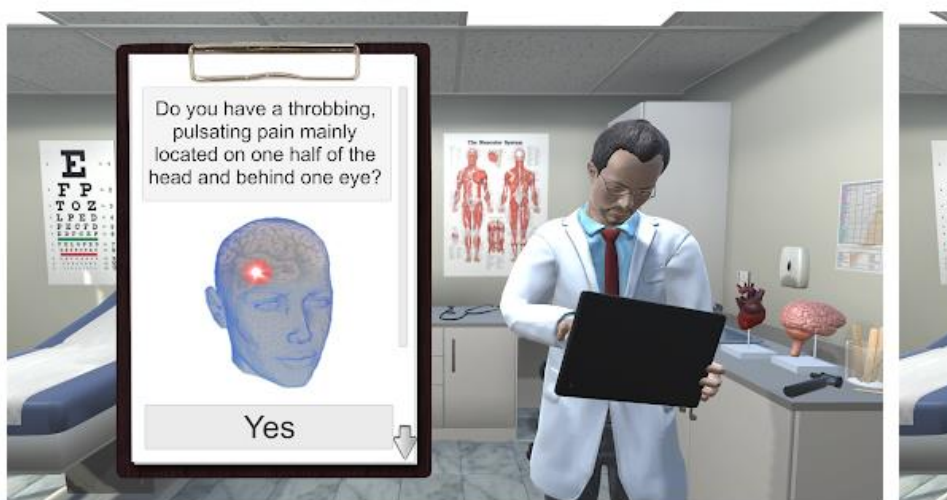


Рисунок 4.2 – Проверка симптомов приложения

Средство проверки симптомов управляется передовым искусственным интеллектом, который синхронизирует ваши симптомы с крупнейшей существующей медицинской базой данных, которая включает более 1500 комбинаций симптомов.

Средство проверки симптомов соответствует последним научным стандартам и регулярно получает обновления, поэтому мы можем

гарантировать, что это приложение соответствует самым высоким медицинским стандартам.

Medical diagnostics – это услуга, которая позволяет пациенту узнать о возможных заболеваниях на основании симптомов, на которые он жалуется и которые поражают его в определенной части тела (рисунок 4.3).

Выбирая симптомы, на которые вы жалуетесь в той или иной части или частях вашего тела, вы можете узнать о возможных заболеваниях, которыми вы страдаете, а также можете больше узнать об этих заболеваниях и узнать их причины и методы лечения.

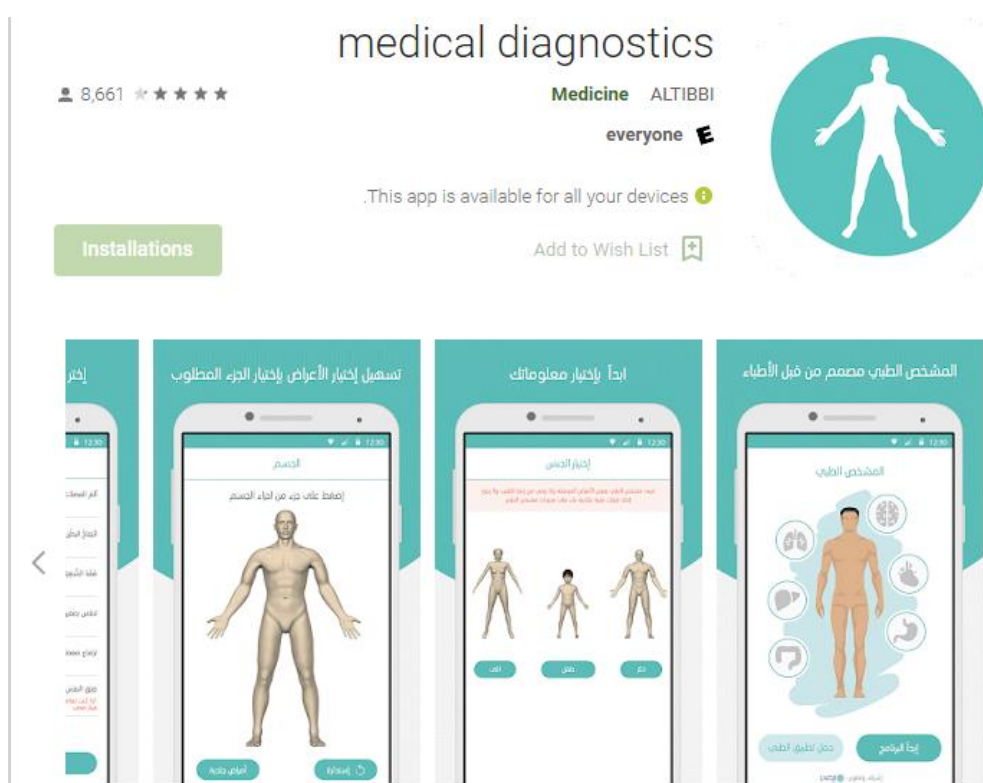


Рисунок 4.3 – Приложение Медицинская диагностика

LetsGetChecked предлагает расширенные возможности тестирования здоровья на дому и доставки рецептов с быстрыми и точными лабораторными результатами, сертифицированными CLIA, и клинической поддержкой 1-на-1 на каждом этапе. Возьмите под контроль свое здоровье с помощью собственной персонализированной панели инструментов, которая

позволяет отслеживать информацию о вашем здоровье и лекарствах (рисунок 4.4).

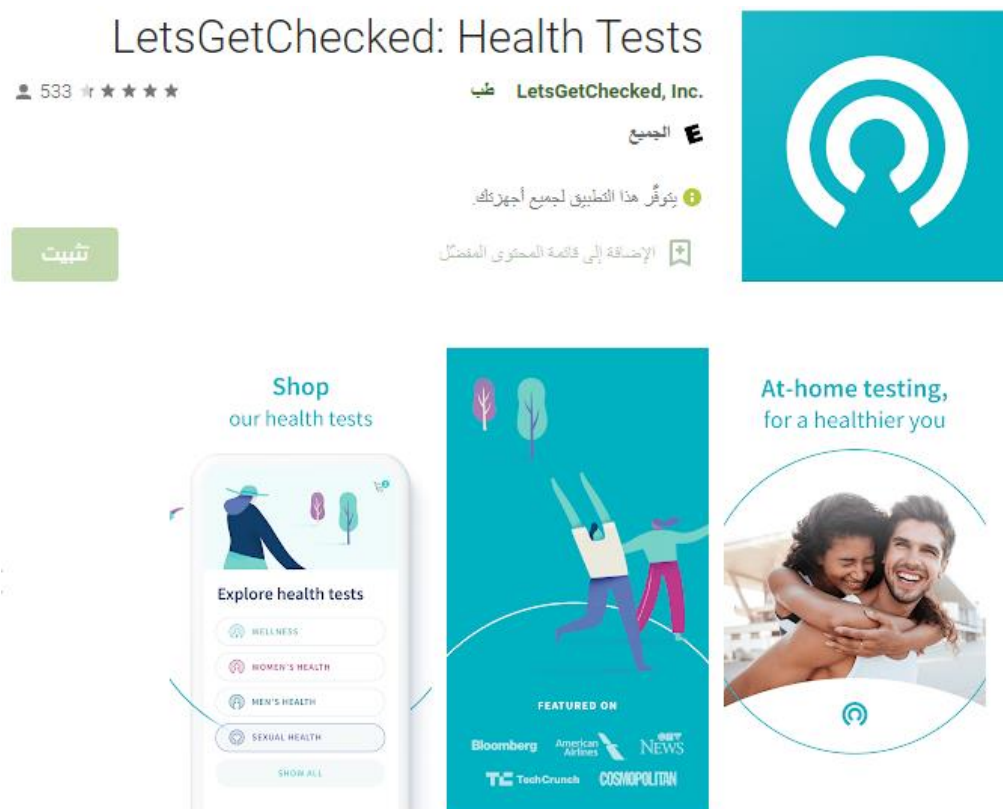


Рисунок 4.4 – Приложение LetsGetChecked

Нет необходимости в клиниках или встречах с LetsGetChecked. Закажите необходимые анализы и лекарства, в том числе противозачаточные средства, и на этих этапах объясняется рабочий механизм приложения:

1. Закажите тест. Закажите тест онлайн с бесплатной доставкой. Все наши тесты доставляются скрытно в простом конверте без видимой ссылки на LetsGetChecked, для личного опыта.

2. Соберите образец. Вы можете забрать образец из дома всего за несколько минут, следуя простым инструкциям. Просто верните образец, используя этикетку с предоплатой, включенную в тест.

3. Получите быстрый результат через 2-5 дней. Ваш образец будет обработан в ультрасовременной лаборатории, сертифицированной CLIA,

которая используется врачами и больницами. Вы можете получить к ним мгновенный доступ через свой безопасный онлайн-аккаунт.

4. Получите медицинскую поддержку. Наши домашние тесты здоровья включают в себя:

Скрининг рака, включая рак толстой кишки, рак предстательной железы и рак шейки матки

Функциональные тесты органов, такие как функциональные тесты почек и печени:

- диабет (HbA1c) и анализ на холестерин;
- информация о питании с помощью тестирования микроэлементов и минералов;
- тестирование мужских гормонов, включая тестостерон;
- тестирование женских гормонов, включая овариальный резерв;
- тестирование щитовидной железы;
- тестирование на сексуальное здоровье (ЗППП);
- тестирование на болезнь Лайма;
- тестирование витаминов, включая витамины B12 и D, фолиевую кислоту и омега-3 и 6.

Тут много разных приложений, но цель не ясна и эта информация используется не в том месте. Благодаря этой информации и аналитике людей мы можем извлечь из нее пользу, а именно:

- узнайте о болезнях, распространяющихся в регионе, прямо сейчас;
- помощь врачам увидеть состояние больного и советы разных врачей о нем;
- оставьте отзыв о пациенте;
- вы можете искать аптеки поблизости;
- посмотреть информацию о пациенте.

Предлагаемая ПП по сравнению с ее аналогами предоставляет более широкий спектр медицинских услуг собранных в единую сеть, также контролируется министерством здравоохранения и гарантирует предоставление компетентной медицинской помощи.

5 Разработка структурной схемы приложения

5.1 Диаграмма сущность-связь

ER – модель (Entity relationship model – модель «сущность – связь») – модель данных, позволяющая описывать концептуальные схемы предметной области.

Характеристика диаграмм «сущность – связь» - данная диаграмма обеспечивает стандартный способ определения данных и отношения между ними, включая сущности и взаимосвязи, отражающие основные бизнес – правила предметной области (рисунок 5.1).

В процессе проектирования базы данных выделены следующие сущности:

- пациент (patient);
- врач (doctor);
- пользователь (User);
- аптека (pharmacy);
- биография (bio);
- история (history);
- фото (photo);
- больница (hospital);
- портфолио (portfolio);
- обзоры (reviews);
- болезнь (illness);
- лекарство (medicine).

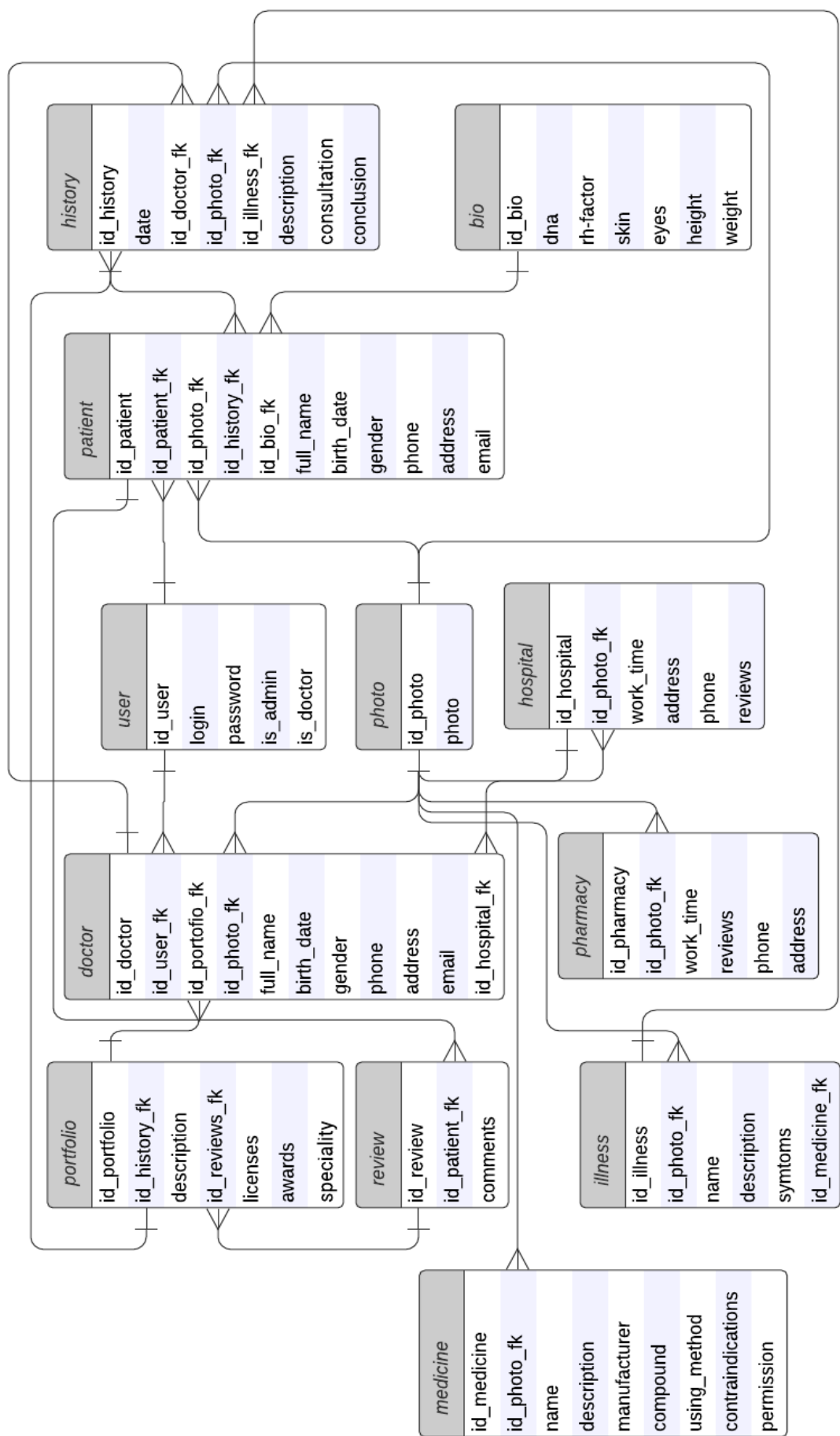


Рисунок 5.1 – Диаграмма сущность-связь

Именно по данной диаграмме можно понять, как работает база данных, а также рассмотреть связи между сущностями. Таблица является главным элементом. На рисунке отображено двенадцать таблиц. В каждой из таблиц есть ключевое поле, через которое проходит связь один ко многим.

5.2 Диаграмма классов

Диаграмма классов – является центральным звеном разработки программного обеспечения.

Класс – совокупность общих признаков заданной группы объектов ПО.

Под отношением классов понимают статическую связь между классами.

На представленном ниже рисунке изображена диаграмма классов ПО (рисунок 5.2).

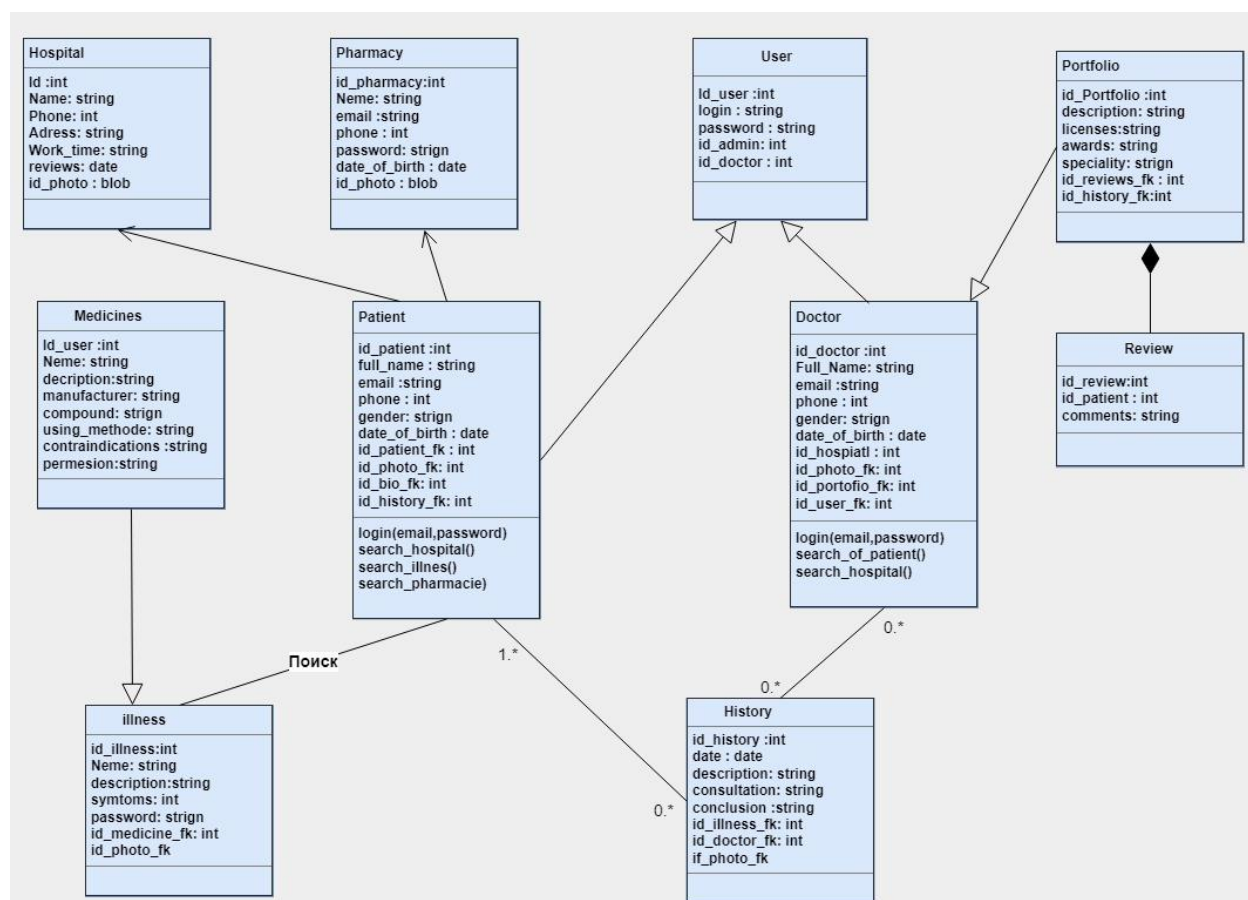


Рисунок 5.2 – Диаграмма классов

На представленном рисунке изображены 7 классов проектируемого программного продукта и связи между ними.

5.3 Диаграмма вариантов использования

Приложение будет обладать обширным набором функций для работы с медиафайлами. На изображение (рисунок 5.3) показана диаграмма вариантов использования, перечисляющая основные функции приложения, которые будут доступны для пользователя.

Диаграмма вариантов использования диаграмма, описывающая, какой функционал разрабатываемой программной системы доступен каждой группе пользователей и в этой системе можно выделить следующие группы пользователей:

- доктор;
- админ;
- пациент.

Каждая из групп пользователей может пользоваться нашей системой по-своему.

Врачи могут:

- смотреть информацию о пациенте;
- предлагайте новые болезни;
- оставить заметку о пациенте;
- управленческий аккаунт;
- посмотреть свою статистику;
- посмотреть статистику заболеваний в регионе;
- контакт с пациентом.

Пациенты могут:

- ищет свою болезнь;
- узнайте о болезнях, распространяющихся в регионе, прямо

сейчас;

- управленческий аккаунт;
- контакт с врачами;
- поиск аптек;
- поиск больниц;
- читайте последние новости и советы о его болезни;
- знание типа и состояния его болезни.

Администраторы могут:

- отправлять отчеты о состоянии здоровья в минздрав региона;
- контрольный аккаунты пациентов;
- контрольный аккаунты врачи;
- добавить места для аптек и больниц;
- добавить болезни и симптомы болезни в базу данных.

Кроме того, у системы есть функционал, который доступен всем группам пользователей. В разрабатываемой нами системе актуально будет добавить мессенджер, в котором можно будет быстро связываться с интересующим человеком.



Рисунок 5.3 – Диаграмма вариантов использования

5.4 Разработка IDEF0 и IDEF3 диаграмм

Основой разрабатываемого нами сервиса является определение заболеваний, определение состояния здоровья человека по имеющимся симптомам. После исследования симптоматики и определения состояния здоровья, сервис (приложение) должен проконсультировать пользователя, дать определенные рекомендации и выдать информацию о лекарствах и их применении. Сервис может использовать различные алгоритмы определения диагнозов, однако, изначально должно быть понимание всего функционала, его взаимодействия между собой. Для составления такой картины ранее были разработаны диаграммы сущность-связь, классов и вариантов использования. В целях полного описания функционала, существует необходимость разработки диаграмм IDEF, активности и последовательности.

IDEF (I-CAM DEFinition или Integrated DEFinition) – методологии семейства ICAM (Integrated Computer-Aided Manufacturing) для решения задач моделирования сложных систем, позволяют отображать и анализировать модели деятельности широкого спектра сложных систем в различных разрезах. При этом широта и глубина обследования процессов в системе определяется самим разработчиком, что позволяет не перегружать создаваемую модель излишними данными. Существует стандарт, который основан на программе интегрированной компьютеризации производства научно-исследовательских лабораторий BBC Райт (ICAM), Часть II, том IV Руководство по функциональному моделированию (IDEF0), Июнь 1981 года. Настоящий стандарт описывает язык моделирования IDEF0 (семантика и синтаксис), а также связанные с ним правила и методы разработки структурированных графических представлений системы или предприятия. Использование стандарта позволяет создавать модели, описывающие системные функции (действия, поведение, процессы, операции), функциональные связи и данные (информация или объекты), которые поддерживают интеграцию систем. Этот стандарт является информационно-

справочным документом, предназначенным для системных или корпоративных разработчиков моделей, применяющих IDEF0 для определения инструментов при реализации этой методики и других компьютерных специалистов для понимания точных синтаксических и семантических правил стандарта.

В процессе разработки программы нашего сервиса был применен этот стандарт, результат применения которого можно увидеть на рисунке 5.4 и рисунке 5.5, Данные диаграммы являются основными информационно-справочными документами.

5.5 Диаграмма активности

Диаграмма активности помогает нам описать логику поведения системы. Можно построить несколько диаграмм активности для одной и той же системы, причем каждая из них будет фокусироваться на разных аспектах системы, показывать различные действия, выполняющиеся внутри нее. На изображение (рисунок 5.6) предоставлена диаграмма активности.

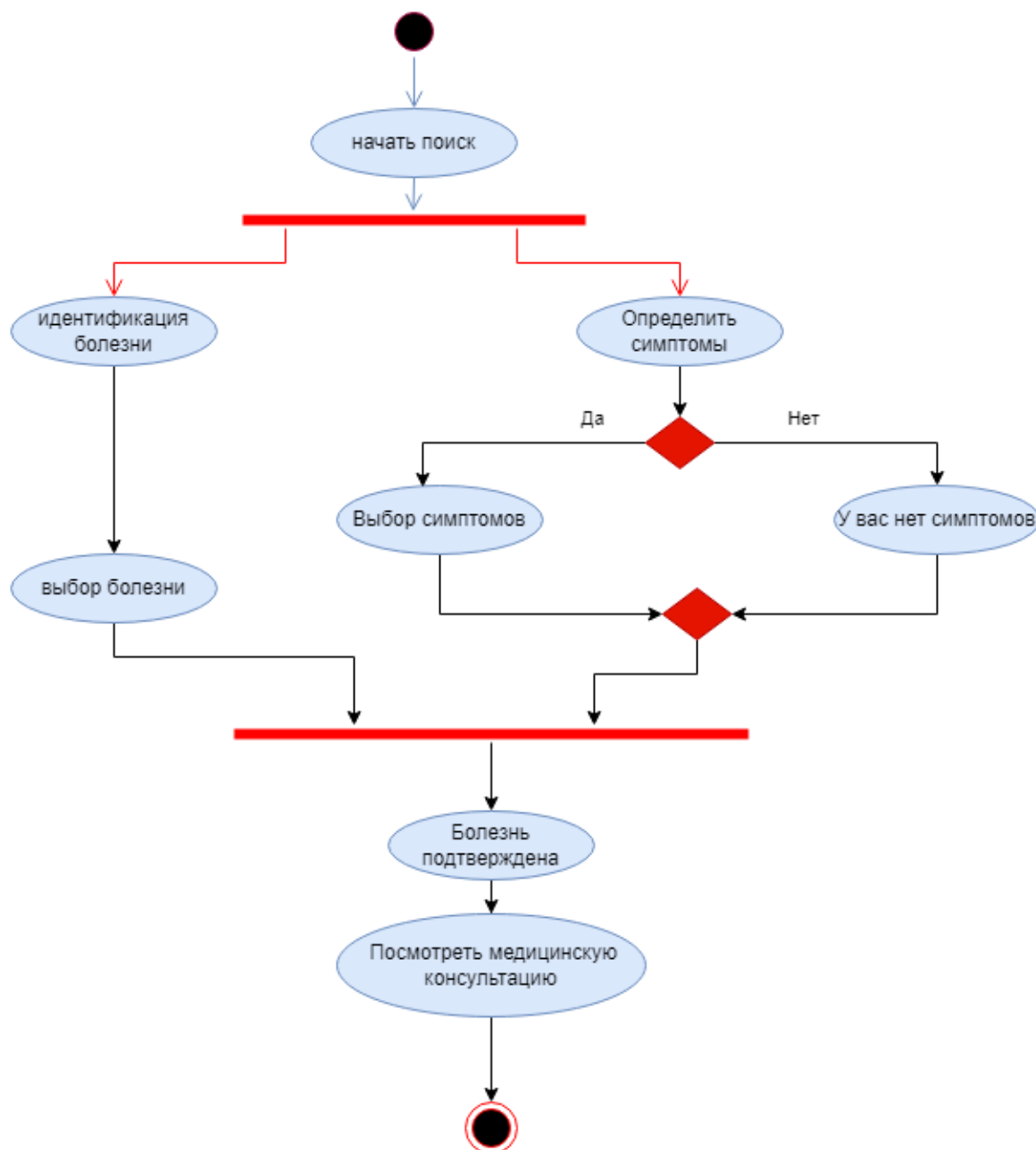


Рисунок 5.6 – Диаграмма активности

5.6 Диаграмма последовательности

Диаграмма последовательности, на которой для некоторого набора объектов на единой временной оси показан жизненный цикл объекта (создание-деятельность-уничтожение некой сущности) и взаимодействие актеров (действующих лиц) информационной системы в рамках прецедента.

На изображении (рисунок 5.7) на диаграмме последовательности показан процесс запроса врачом информации о пациенте из системы.

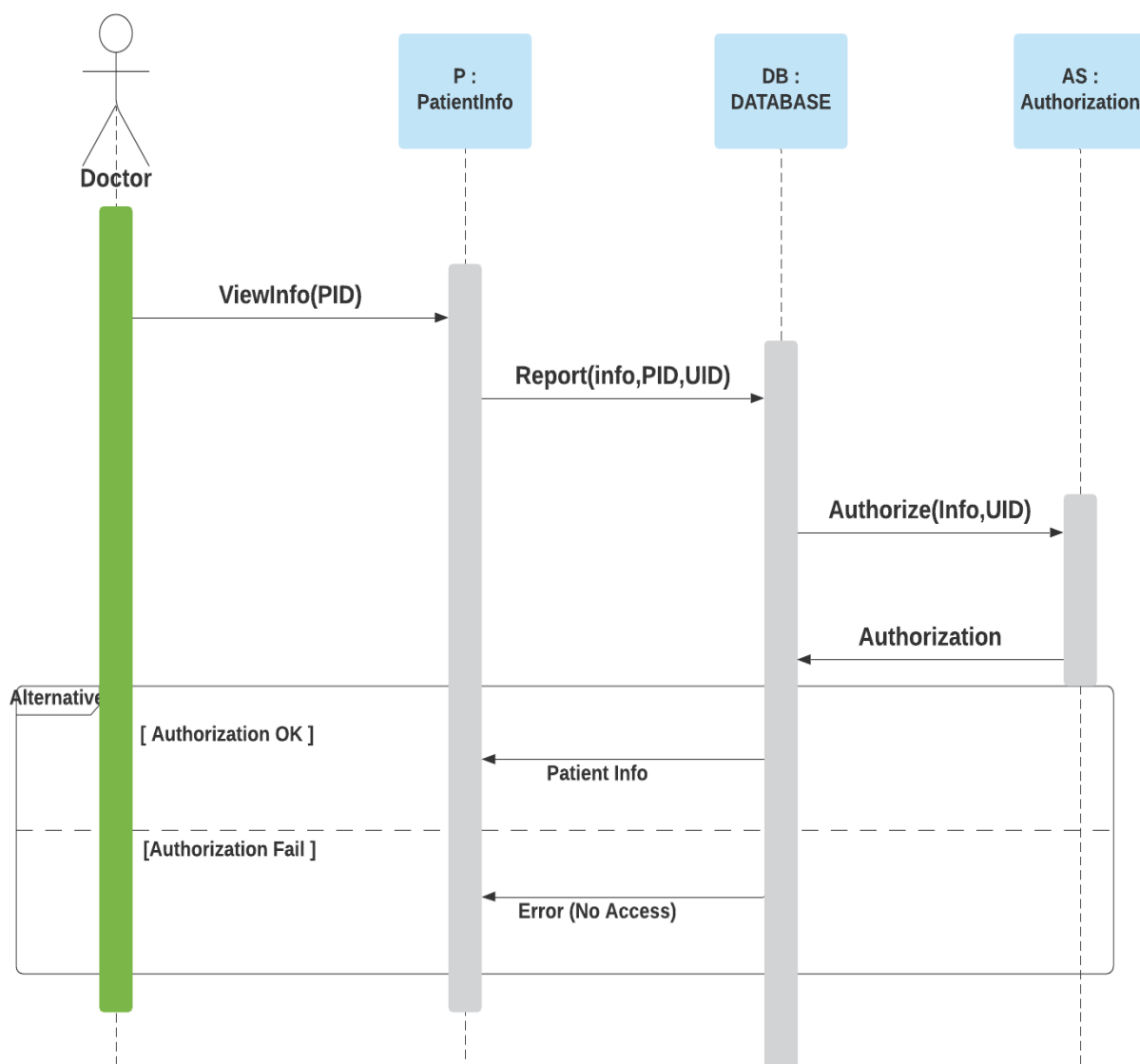


Рисунок 5.7 – Диаграмма последовательности для функции получения информации

Первым шагом является поиск пациента, а затем отправление отчета в базу данных, после чего необходимо убедиться, что у пользователя есть доступ к информации, и у него будет два варианта:

- правильная проверка и отображение информации;
- проверка на ошибки и недостающую информацию.

В результате разработки мы смогли описать состав разрабатываемого сервиса функциональной диагностики, получить полную картину всего программного обеспечения. Разработанные диаграммы позволят разработать программный продукт без лишней траты времени.

6 Реализация приложения

6.1 Предварительные установки

Для разработки данной программы первым делом нужно определиться на каком платформе будет разрабатываться мобильной приложения с функциональной диагностикой.

Если выбрать Android Studio, тогда её нужно скачать с официального сайта <https://developer.android.com/studio>.

так или иначе нужно скачать также java jdk, её можно скачать с официального сайта <https://www.oracle.com/java/technologies/downloads/>.

нам также понадобится Android SDK Platform-Tools – это компонент для Android SDK. Он включает в себя инструменты, взаимодействующие с платформой Android, в первую очередь adb и fastboot. разработчики приложений обычно просто используют установки копии Studio. Эта загрузка полезна, если вы хотите использовать adb непосредственно из командной строки и у вас не установлена Studio. Его можно скачать с официального сайта: <https://developer.android.com/studio/releases/platform-tools>.

6.2 Разработка функциональной модели

Основой описанного выше сервиса станет, конечно же, именно определение заболеваний, определение состояния здоровья человека по имеющимся симптомам. После исследования симптоматики и определения состояния здоровья, сервис (приложение) должен проконсультировать пользователя, дать определенные рекомендации и выдать информацию о лекарствах и их применении. Сервис может использовать различные алгоритмы определения диагнозов, имея базу данных с такими диагнозами и соответствующими симптомами. В случае, если база данных будет, мы возможно применение обычных, известных каждому программисту,

алгоритмов ветвления (в Си-подобных языках – «if – else», а также «switch – case»). Однако, предполагается, что база данных с количеством симптомов, соответствующими болезнями, недугами, рекомендациями и прочим будут обширными и, при этом, постоянно пополняемыми. Дело в том, что врачебное дело и наука не стоят на месте, появляются все новые способы борьбы с болезнями, новые рекомендации. К сожалению, появляются и новые болезни. В связи с этим, предполагается, что в рамках программного обеспечения сервиса должен быть разработан искусственный интеллект. В процессе долгих теоретических поисков и экспериментов, было выяснено, что наиболее подходящей основой его реализации должно стать уже давно известное машинное обучение (Machine Learning) – класс методов искусственного интеллекта, особенностью которых является не прямое решение задачи, а обучение за счёт использования решений множества сходных задач. Благодаря машинному обучению программист не будет писать инструкции, учитывающие все возможные проблемы и содержащие все решения. Вместо этого в программное обеспечение нашего сервиса будет заложен алгоритм самостоятельного нахождения решений путём комплексного использования статистических данных, из которых выводятся закономерности и на основе которых делаются прогнозы. Мы пришли к выводу, что наиболее соответствующим в нашем случае является обучение с учителем (Supervised Learning) – это раздел машинного обучения, который ориентирован на решение задачи, которая как раз соответствует нашей. Сама задача в общем смысле описана далее.

Naïve Bayes Classifier Algorithm (Алгоритм наивного байесовского классификатора) – это алгоритм обучения с учителем (*Supervised Learning*), который основан на теореме Байеса и используется для решения задач классификации.

Наивный байесовский классификатор один из простых и наиболее эффективных алгоритмов классификации, который помогает создавать быстрые модели машинного обучения, способные делать быстрые прогнозы,

вероятностный классификатор, что означает, что он предсказывает на основе вероятности объекта (рисунок 6.1).

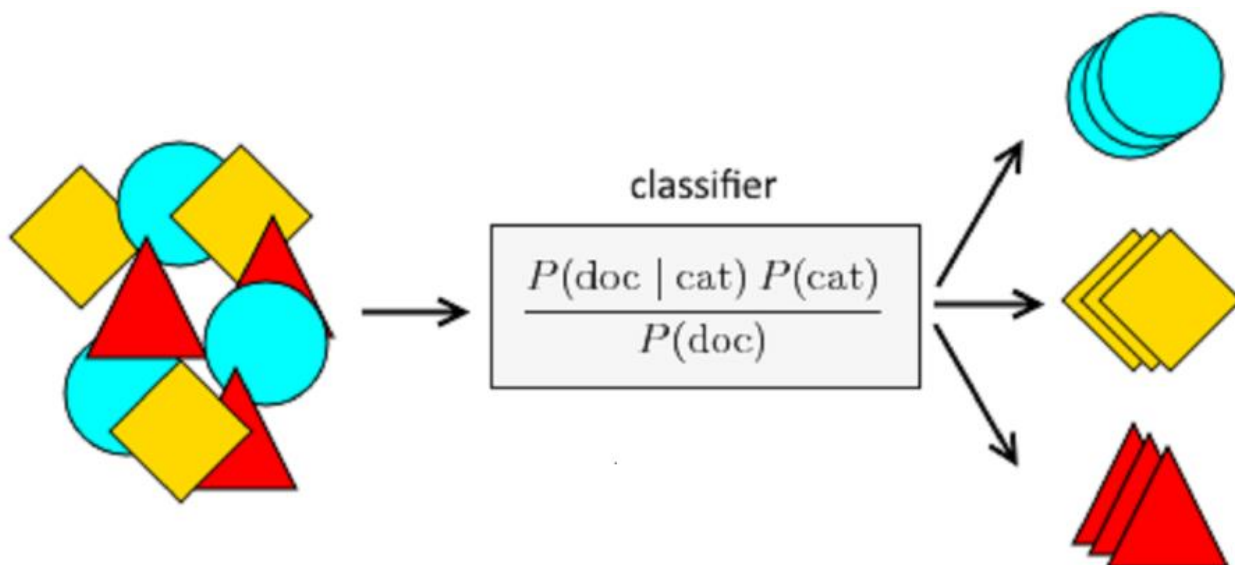


Рисунок 6.1 – Разработка алгоритма наивного байесовского классификатора

Некоторыми популярными примерами наивного байесовского алгоритма являются фильтрация спама, сентиментальный анализ и классификация статей.

Наивный: он называется наивным, потому что предполагает, что появление определенной функции не зависит от появления других функций. Например, если фрукт идентифицируется по цвету, форме и вкусу, то красный, сферический и сладкий фрукт распознается как яблоко. Следовательно, каждый признак по отдельности способствует идентификации того, что это яблоко, независимо друг от друга.

6.3 Разработка технической и справочной информации

Создание первого apk приложения В Android Studio нам нужно создать виртуальное устройство Android (AVD), которое эмулятор может использовать для установки и запуска нашего приложения. или мы можем подключить наш мобильный телефон к устройству и запустить.

Когда мы запустим программу, мы заметим внешний интерфейс приложения, который объясняет условия работы приложения для пользователя (рисунок 6.2).

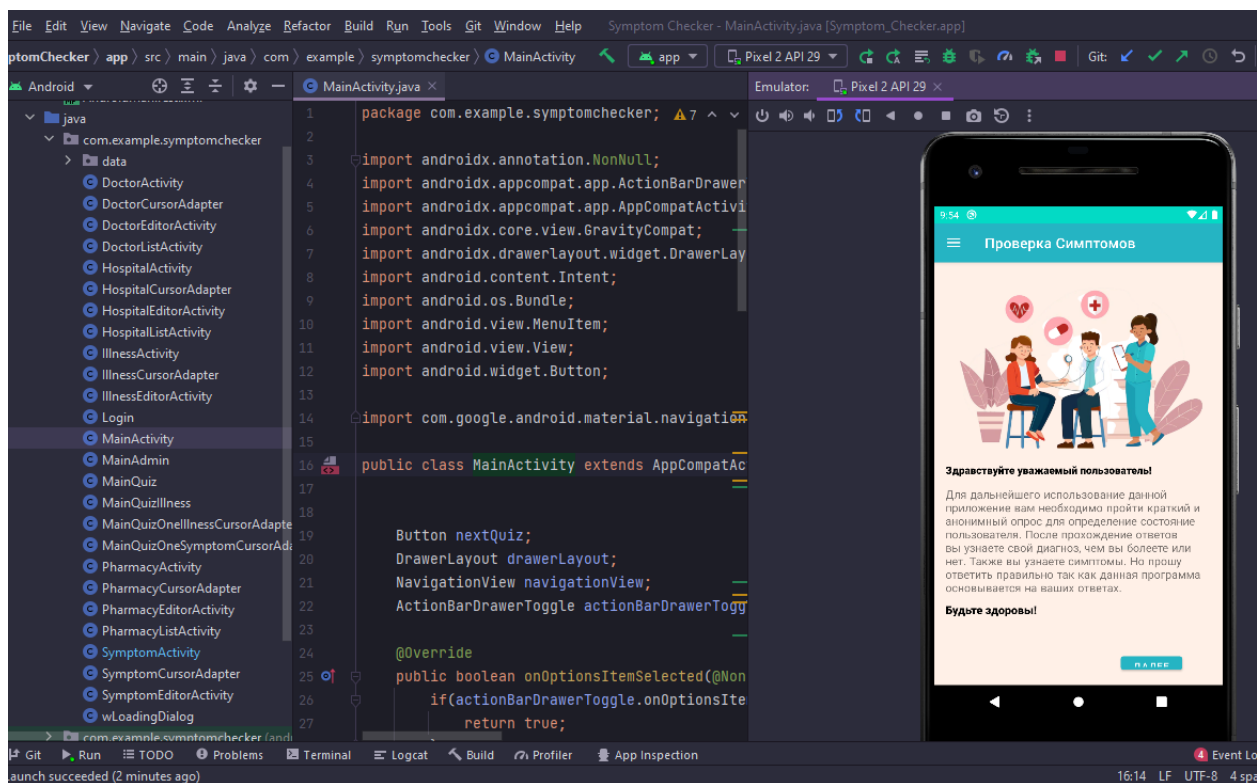


Рисунок 6.2 – интерфейс приложения.

Приложение разделено на две части

- административная часть;
- часть пациента.

Административная часть связана с управлением базой данных (добавление, изменение, удаление), который представлен в:

- управление списком болезней;
- управление списком симптомов болезни;
- управление списком больниц;
- управление списками аптек;
- управление списком врачей.

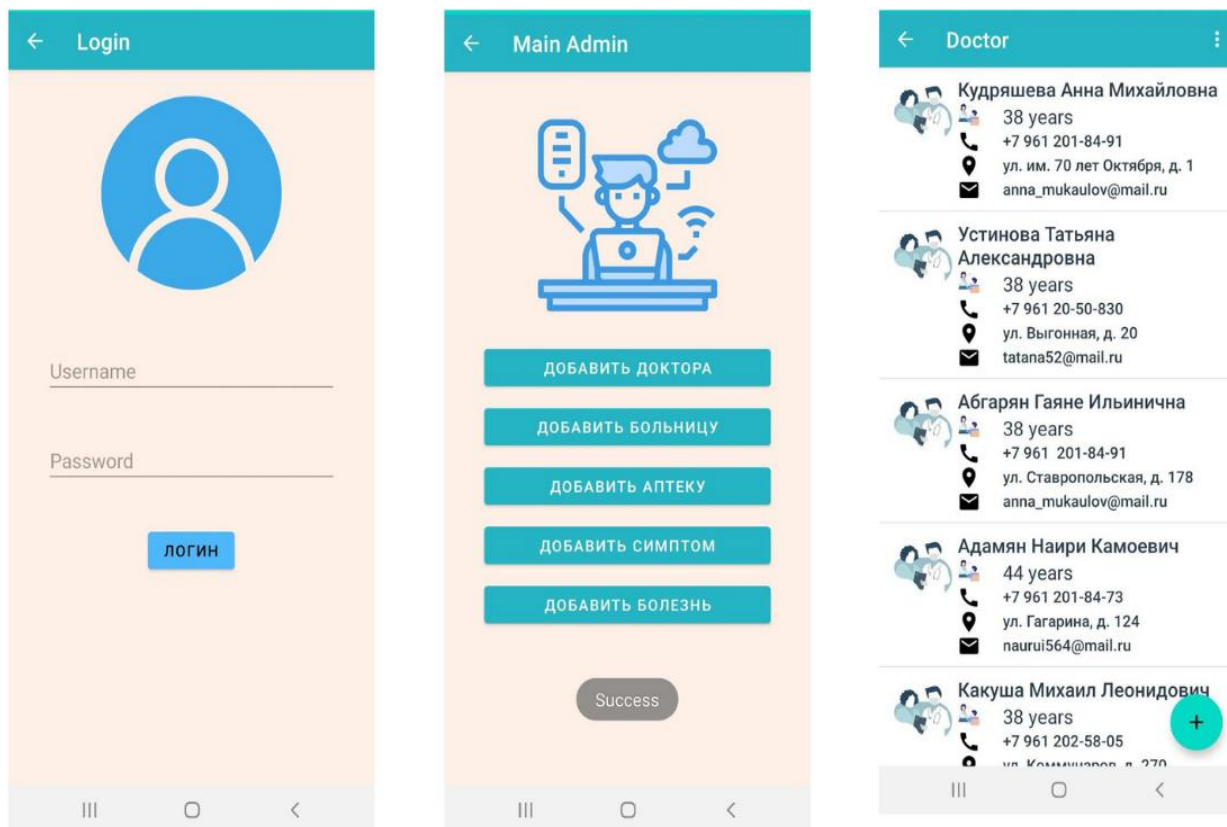


Рисунок 6.3 – Администратор интерфейса

В разделе пациента диагноз пациента будет поставлен двумя способами (рисунок 6.4):

- путем диагностики его болезни с помощью набора вопросов;
- или предполагая болезнь.

вопрос к пациенту (минутку)



Пожалуйста, ответьте на следующие вопросы

- ☒ Диагностируя свое заболевание
- ☐ Я знаю тип болезни

ДАЛЕЕ

Рисунок 6.4 – Интерфейс пациента

В первой части он ответит на вопросы, чтобы иметь возможность определить свое заболевание и показать правильные результаты (рисунок 6.5).

←

Проверка Симптомов

Пожалуйста, проверьте все приведенные ниже утверждения, которые относятся к вам .

выберите один ответ в каждой строке.

у меня высокая температура

☒ Да

☐ Нет

☐ Не знаю

у меня избыточный вес или ожирение

☒ Да

☐ Нет

☐ Не знаю

я курю сигареты

☒ Да

Рисунок 6.5 – По диагностике пациента

В результате у нас есть некоторая рекомендуемая информация (рисунок 6.6) :

- нужна ли экстренная помощь;
- классификация;
- что рекомендуется и не рекомендуется делать;
- возможные причины;
- списки врачей и больниц, аптек.







<p>← Функциональная диагностика</p> <p>Полученные результаты рекомендация Проконсультируйтесь с врачом</p>  <p>Ваше заболевание по предыдущим анализам:</p>  <p>Нужна ли экстренная помощь : Экстренная помощь не требуется</p> <p>Классификация :</p> <ul style="list-style-type: none"> • Первичные запоры (врожденные или приобретенные аномалии толстой кишки). • Вторичные запоры (развиваются в результате заболевания, травмы). • Идиопатические запоры (нарушение моторики кишечника без установленной причины) <p><u>Что рекомендуется делать :</u></p>	<p>← Функциональная диагностика</p> <p>Что рекомендуется делать :</p> <ul style="list-style-type: none"> • Вести активный образ жизни. • Увеличить объем потребляемой жидкости до 1,5-2 литров в день. • Не пропускать приёмы пищи. • Есть продукты, богатые клетчаткой (овощи и фрукты, крупы). <p>Что не рекомендуется делать :</p> <ul style="list-style-type: none"> • Ограничивать приём жидкости и пищи. • Принимать слабительные препараты без назначения врача. <p>Возможные причины :</p> <ul style="list-style-type: none"> • Неспецифические воспалительные заболевания кишечника. • Первичный билиарный холангит. • Синдром раздраженного кишечника. • Новообразование толстой кишки. • Геморрой. • Кишечная непроходимость. • Острый панкреатит. • Хронический холецистит. • Кишечная инфекция. • Хронический панкреатит. • Целиакия. • Гипотиреоз. 	<p>← Функциональная диагностика</p> <p>• Гипотиреоз.</p> <p>Список врачей в вашем районе</p>  <p>Список больниц в вашем районе</p>  <p>Список аптек в вашем районе</p>  <p><small>Обратите внимание, что информация, предоставляемая в этом инструменте, предоставляется исключительно в образовательных целях и не является квалифицированным медицинским заключением. Эта информация не должна считаться советом или мнением врача или другого медицинского работника о вашем фактическом состоянии здоровья, и вам следует обратиться к врачу по поводу любых симптомов, которые могут у вас возникнуть. Если вы испытываете острую медицинскую помощь, вам следует немедленно позвонить по местному номеру службы экстренной помощи, чтобы запросить неотложную медицинскую помощь.</small></p>
---	--	--

Рисунок 6.6 – результат диагностики

Второй способ в том случае, если он знает о болезни и хочет прочесть советы и тяжесть своего заболевания, и наиболее важные шаги, которые он должен выполнить (рисунок 6.7).

← Проверка Симптомов

Вы можете прочитать несколько советов о Вашей болезни



Выбрать болезнь
Пожалуйста, выберите Ваше заболевание, чтобы узнать о нем больше:

Болезнь Альцгеймера

Головная боль напряжения

Миопия

Гиперметропия

Туберкулез

← Проверка Симптомов

Советы и рекомендации по этому заболеванию :

Болезнь Альцгеймера
Нервная система
описание :
Болезнь Альцгеймера — наиболее распространенная форма приобретенного слабоумия (деменции), являющаяся проявлением длительно текущего нейродегенеративного процесса.

Причины :
Заболевание получило название в честь немецкого психиатра, Алоиса Альцгеймера, который впервые его описал.


Существует несколько конкурирующих гипотез, которые объясняют возможные причины развития Альцгеймера. К наиболее популярным относят

← Проверка Симптомов


назначения этих препаратов:

- Противопаркинсонические препараты при БА применяют для коррекции поведенческих расстройств.


Список врачей в вашем районе



Список больниц в вашем районе



Список аптек в вашем районе



Обратите внимание, что информация, предоставляемая этим инструментом, предоставляется исключительно в ознакомительных целях и не является медицинской рекомендацией.

Рисунок 6.7 – Список болезней

Также после анализа и постановки диагноза пациенту предлагается помощь путем подсказки врачей для связи с ними и ближайшими к нему больницами и аптеками (рисунок 6.8).

Проверка Симптомов



Кудряшева Анна Михайловна
38 years
+7 961 201-84-91
ул. им. 70 лет Октября, д. 1
anna_mukaulov@mail.ru

Проверка Симптомов



Центр грудной хирургии
08:00 - 22:00
+79875554132
ул. Российская, 140

Проверка Симптомов



Дешёвая аптека
08:00 - 20:00
88612900722
ул. 40-летия Победы, 33/1

Проверка Симптомов




Устинова Татьяна Александровна
38 years
+7 961 20-50-830
ул. Выгонная, д. 20
tatana52@mail.ru

Проверка Симптомов



Краевая больница №1 им
08:00 - 22:00
+7(861) 252-73-41
ул. 1 Мая, д. 167

Проверка Симптомов



Аптека 112
08:00 - 20:00
88612526555
ул. 40-летия Победы, 108

Проверка Симптомов



Абгарян Гаяне Ильинична
38 years
+7 961 201-84-91
ул. Ставропольская, д. 178
anna_mukaulov@mail.ru

Проверка Симптомов



Инфекционная больница
08:00 - 18:00
+79875554132
ул. Седина, 204

Проверка Симптомов



Аптеки Кубани, № 6
08:00 - 20:00
+7987555413
ул. Рашпилевская, 183

Проверка Симптомов



Адамян Наири Камоевич
44 years
+7 961 201-84-73
ул. Гагарина, д. 124
naurui564@mail.ru

Проверка Симптомов



Городская больница №3 (ХБК)
08:00 - 22:00
+79875554132
ул. Айвазовского, д. 97

Проверка Симптомов



Аптека ВИТА ЦЕНТРАЛЬНАЯ
08:00 - 20:00
+7987555413
Краснодар улица имени, ул. 40-летия Победы, 144/4

Рисунок 6.8 – Список больниц, аптек и врачей

7 Информационная безопасность

7.1 Построение диаграммы потоков данных

Для решения задач по моделированию опасностей разработанной проекте "Мобильное приложение для функциональной диагностики" были построены DFD диаграммы, которые позволяют проанализировать внешние источники и адресаты данных, логические функции, потоки и хранилища данных, к которым осуществляется доступ. Данные диаграммы являются основным средством моделирования функциональных требований к проектируемой системе. С их помощью эти требования представляются в виде иерархии функциональных компонентов, связанных потоками данных.

На рисунке 7.1 представлена контекстная диаграмма для разработанной проекте "Мобильное приложение для функциональной диагностики", также на рисунке 7.2 представлена DFD диаграмма уровня 1 и на рисунке 7.3 представлена схема расположения компонентов.

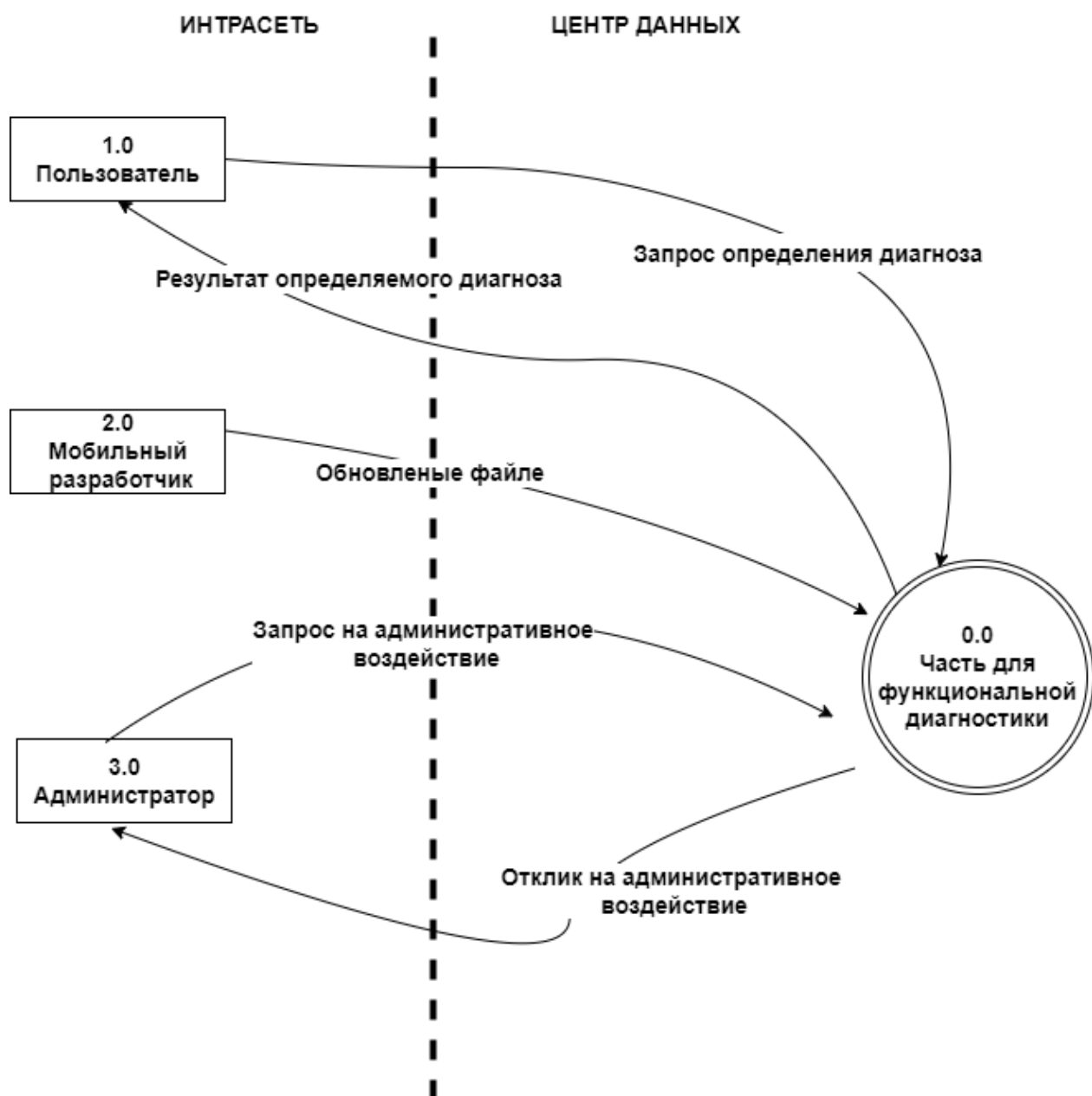


Рисунок 7.1 – Начальная контекстная диаграмма

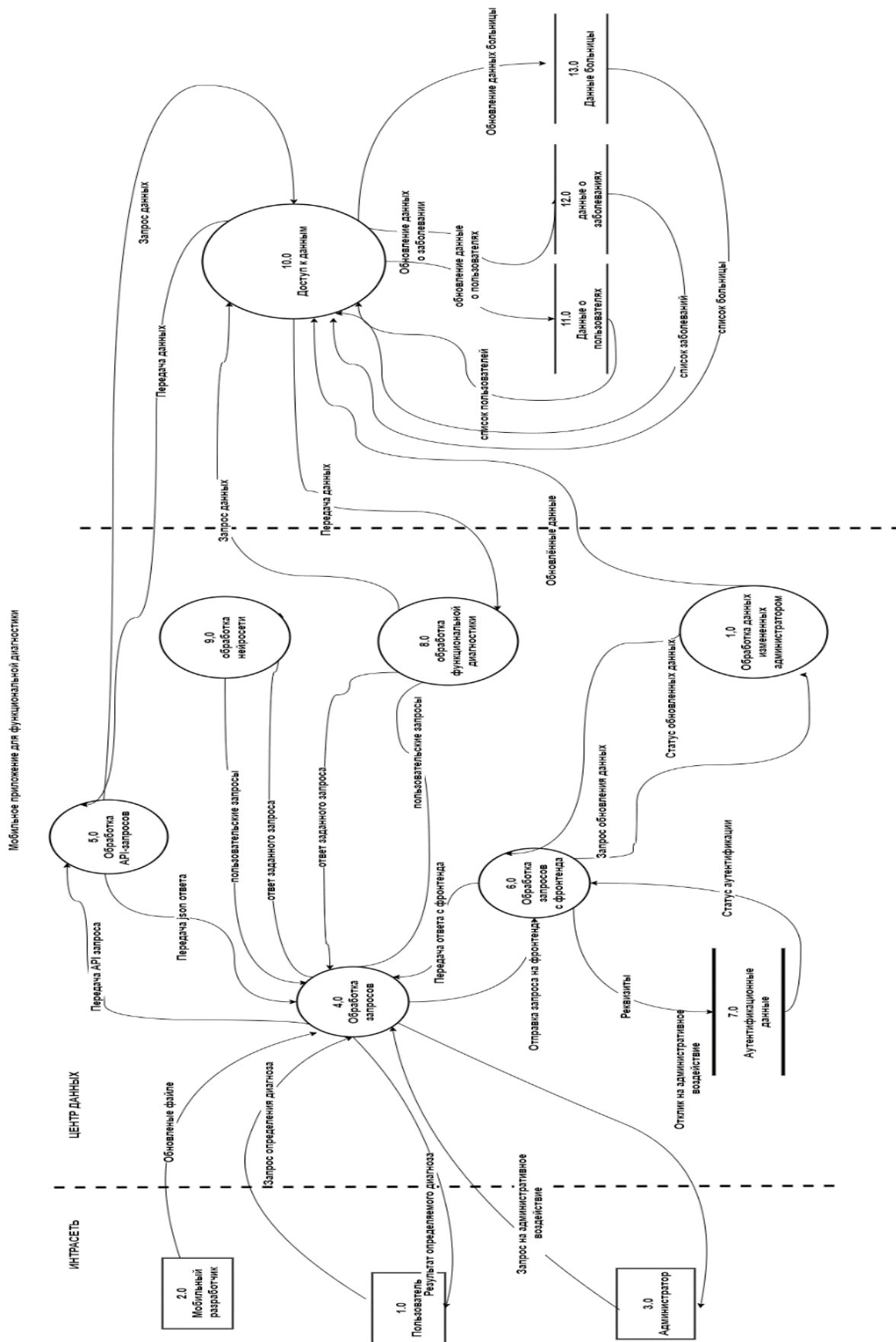


Рисунок 7.2 – DFD диаграмма уровня 1

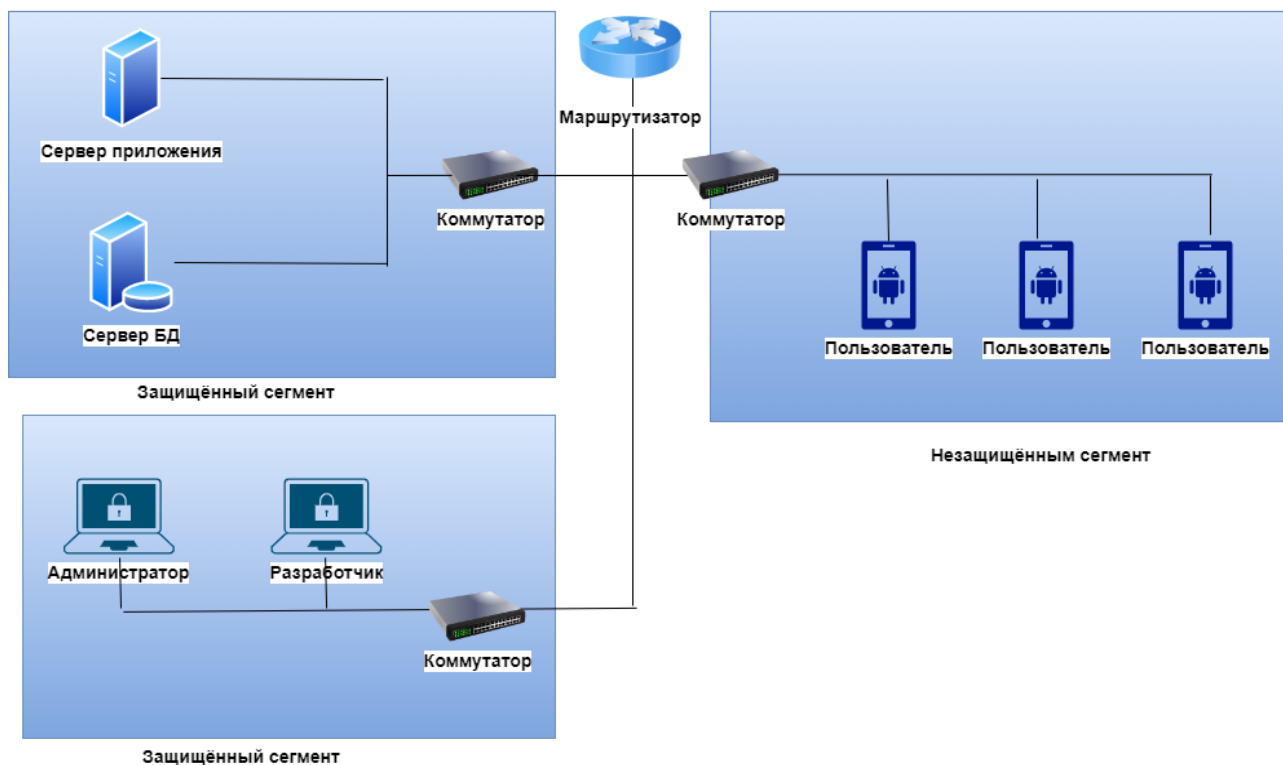


Рисунок 7.3 – Схема расположения компонентов ИС

7.2 Выявление перечня угроз

Помимо построения DFD диаграмм, в данном дипломном проекте, был проанализирован набор опасностей определенный по методике STRIDE. Весь перечень опасностей представлен в таблице 7.1.

Таблица 7.1 – Перечень опасностей STRIDE

Элемент DFD	Подмена (S)	Несанкционированный доступ (T)	Отказ от факта получения или отправки сообщения (R)	Раскрытие информации (I)	Отказ в обслуживании (D)	Повышение привилегий (E)
Внешние сущности						

Продолжение таблицы 7.1

Пользователь		+				+
Мобильный разработчик		+				+
Администратор		+				+
Процессы обработки данных						
Обработка запросов				+		
Обработка API-запросов			+			
Обработка запросов с фронтенда			+			
Обработка нейросети				+		
Обработка функциональной диагностики				+		
Обработка данных измененных администратором				+		
Доступ к данным				+		
Потоки данных						
Запрос определения диагноза			+			
Запрос на административное Воздействие			+			
Отклик на административное Воздействие			+			
Обновленные файле			+			
Передача API запроса			+			
Передача json ответа				+		
пользовательские запросы				+		
ответ заданного запроса				+		
ответ заданного запрос			+			
пользовательские запросы			+			
Передача ответа с фронтенда				+		
Запрос обновления данных				+		
Статус обновленных данных				+		
Запрос данных			+			

Продолжение таблицы 7.1

Передача данных						
обновление данные о пользователях				+		
список пользователей				+		
Обновление данных о заболевании	+					
список заболеваний	+					
Обновление данных больницы	+					
список больницы	+					
Хранилища данных						
Аутентификационные данные	+					
Данные о пользователях	+					
данные о заболеваниях	+					
Данные больницы	+					

7.3 Построение деревьев опасностей

Выполним построение деревьев развития опасностей для пяти произвольных угроз из таблицы 7.1. На рисунка 7.4 – 7.8 показаны деревья 5 произвольных опасностей разрабатываемой информационной системы.

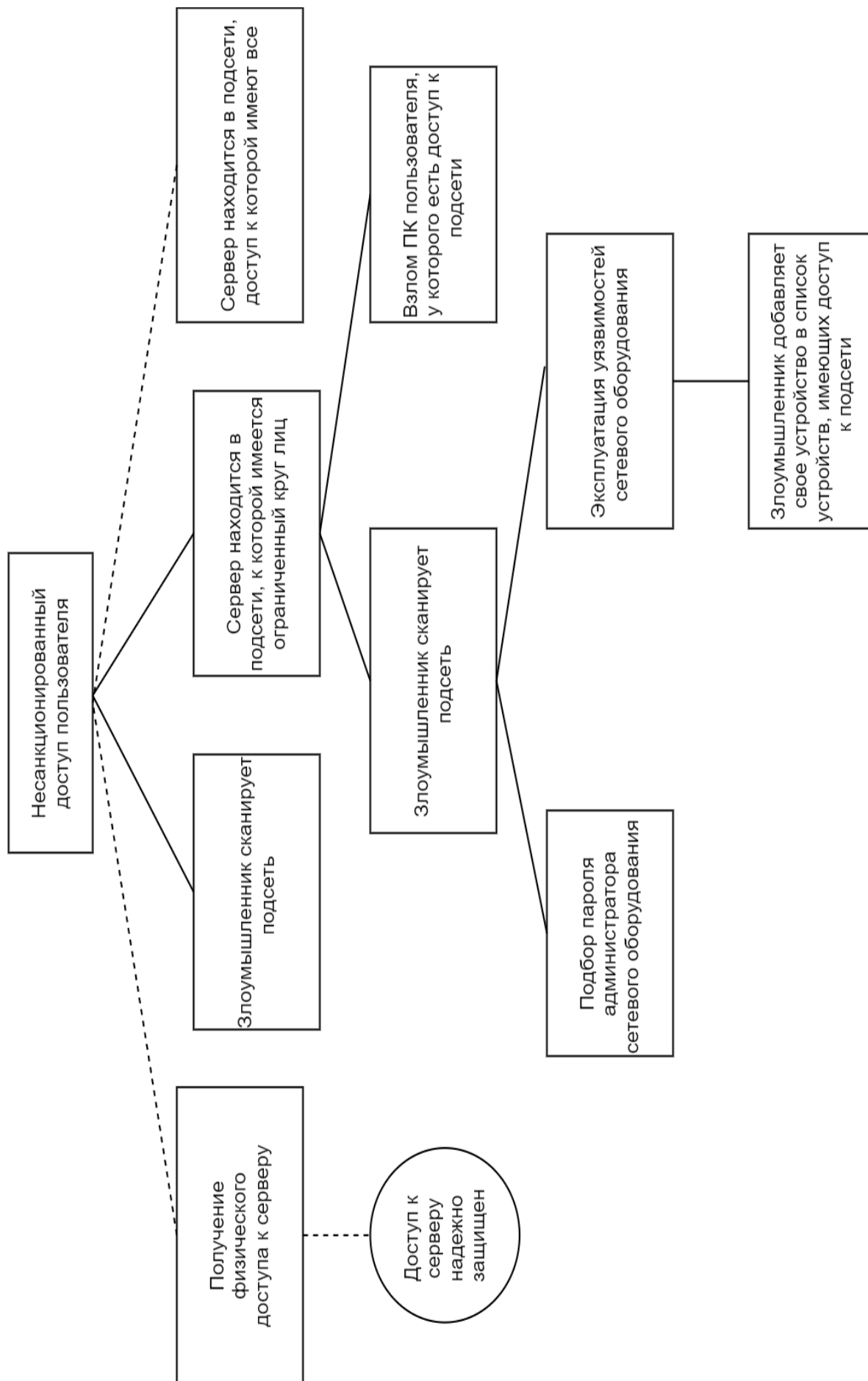


Рисунок 7.4 – Дерево опасности несанкционированного доступа пользователя

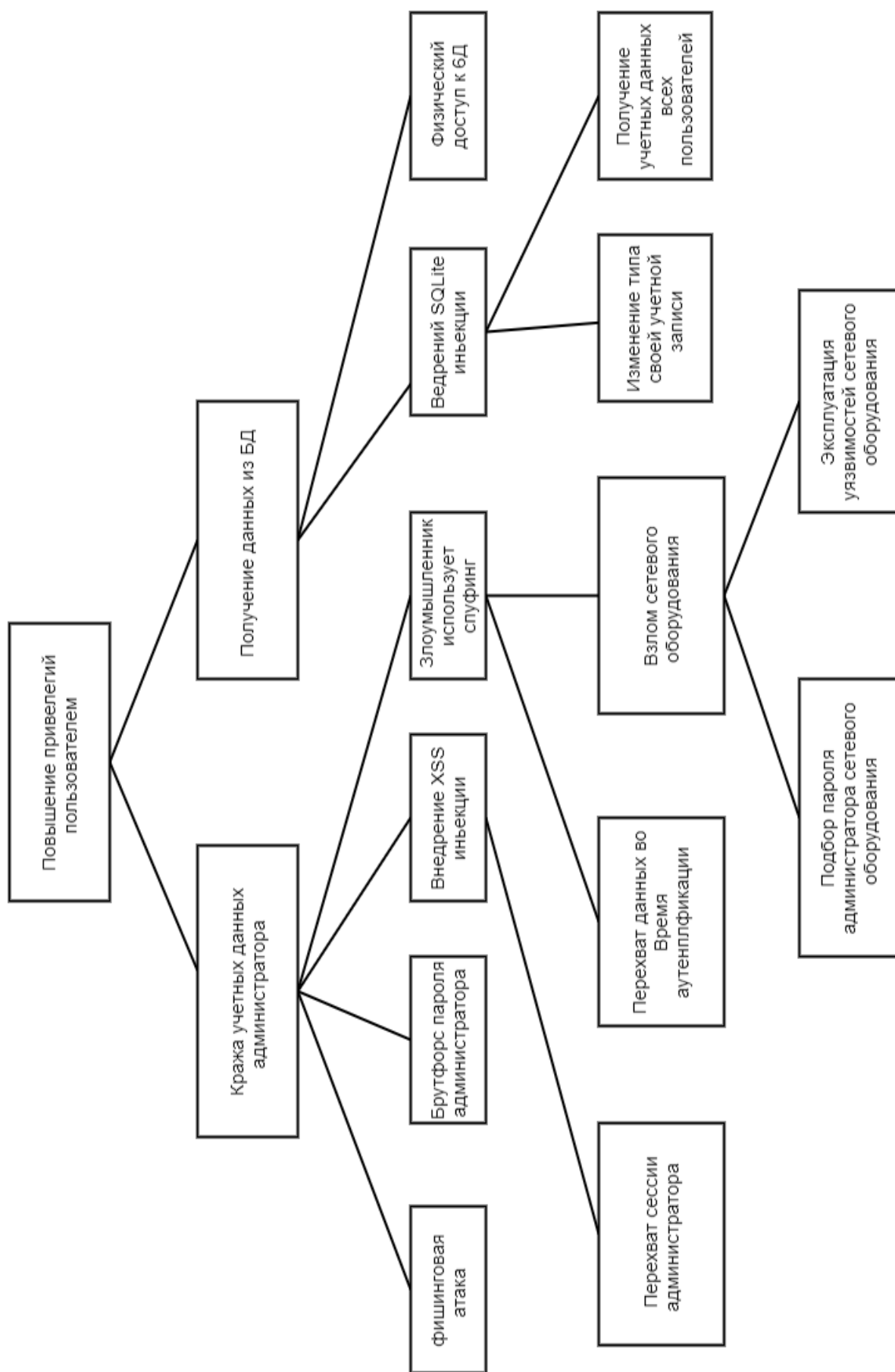


Рисунок 7.5 – Древо опасности повышения привилегий пользователем

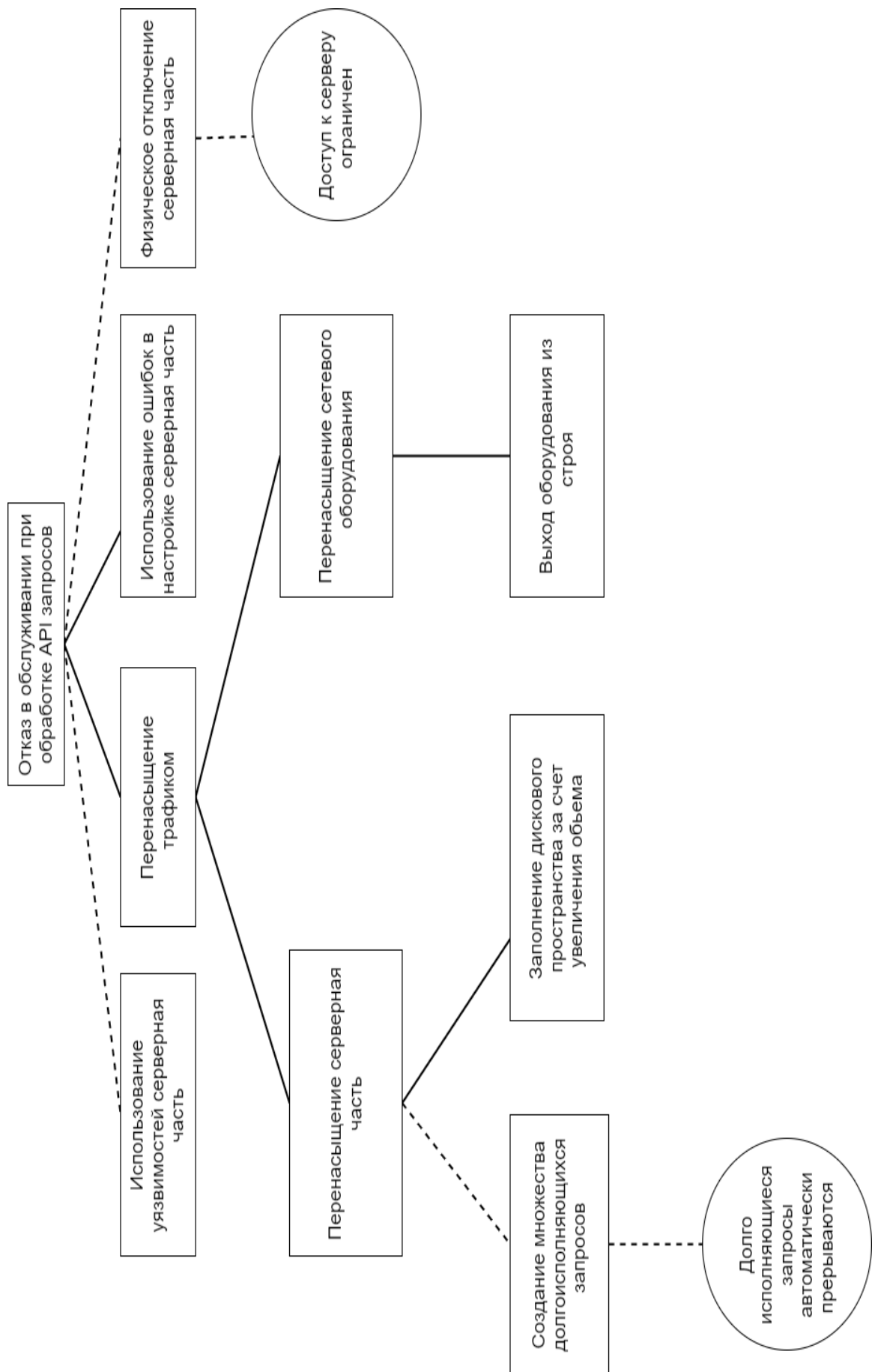


Рисунок 7.6 – Дерево опасности отказа в обслуживании при обработке API-запросов

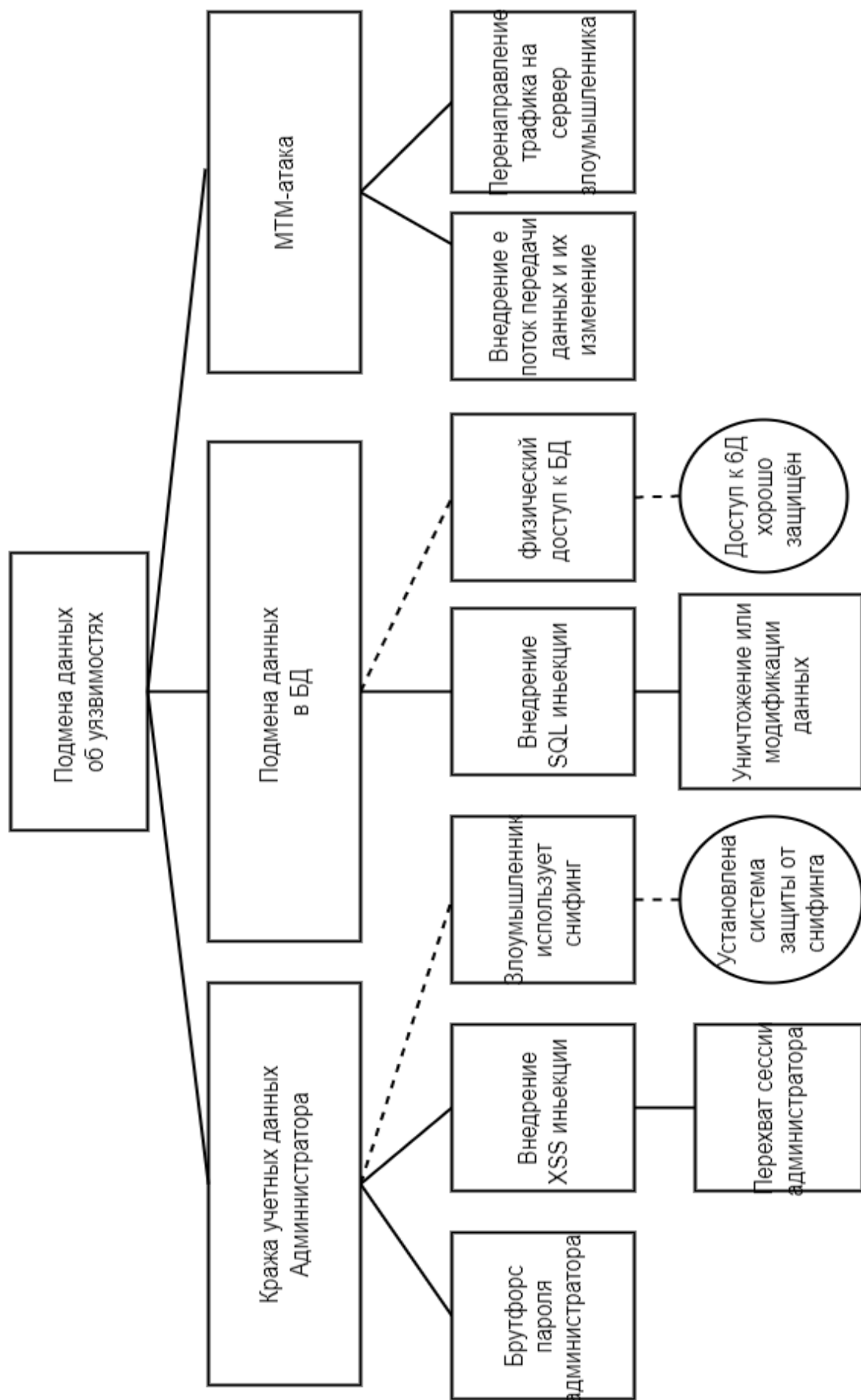


Рисунок 7.7 - Дерево опасности подмены данных об уязвимостях

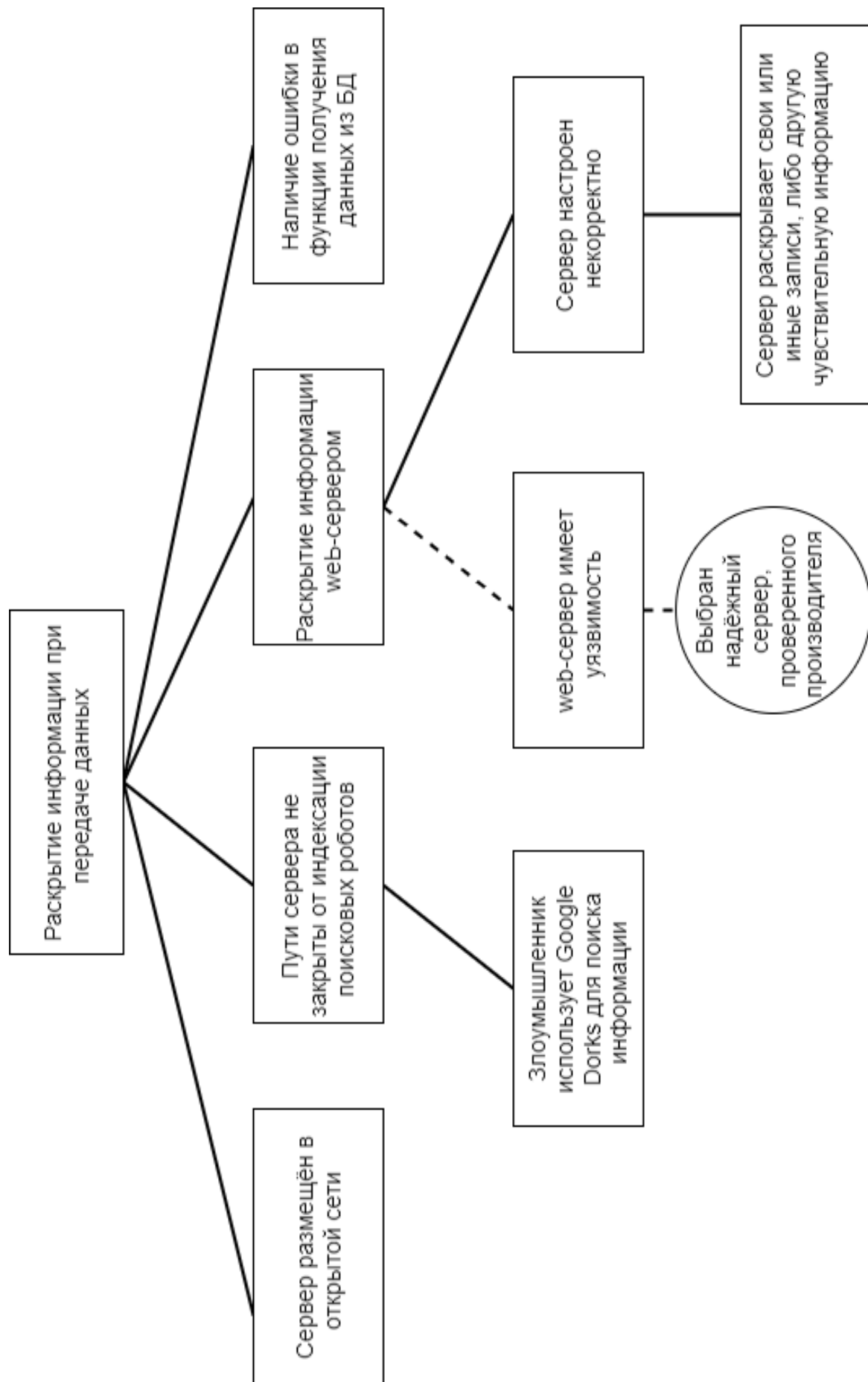


Рисунок 7.8 – Дерево опасности раскрытия информации при передаче данных

7.4 Анализ рисков опасности

В таблице 3 выполним расчет рисков разработанной информационной системы по методу DREAD для выбранных ранее опасностей.

Данная методика расчета производит оценку риска по определённым показателям: потенциальный ущерб, повторяемость, простота организации атаки, затронутые пользователи, простота обнаружения. Каждый критерий оценивается по шкале от 1 до 10 баллов. Затем итоговая оценка производится с использованием уравнения 7.2:

$$RISK_{DREAD} = \frac{D+R+E+A+D}{5} \quad (7.1)$$

Таблица 7.2 – Расчет оценки рисков по методу DREAD

Опасность	<i>D</i>	<i>R</i>	<i>E</i>	<i>A</i>	<i>D</i>	Итоговая оценка риска
Повышение привилегий пользователем	7	5	7	6	9	6,8
Несанкционированный доступ пользователя	9	6	8	9	8	8,0
Отказ в обслуживании при обработке API - запросов	9	7	5	9	9	7,8
Подмена данных об уязвимостях	8	5	6	9	8	7,2
Раскрытие информации при передаче данных	8	5	7	8	9	7,4

Из таблицы 7.2 видно, что для разработанной в выпускной квалификационной работе наибольшую опасность представляет угроза несанкционированного доступа пользователя.

7.5 Меры по предотвращению опасностей

Для предотвращения несанкционированного доступа пользователя серверную часть приложения следует выполнять ряд требований:

- сетевое оборудование, к которому подключается серверная часть приложения, должно иметь функционал обнаружения сетевых атак;
- база данных должна быть защищена, и пользователи не должны иметь к ней доступ;
- приложение должно постоянно обновляться;
- сетевое оборудование должно регулярно обновляться до последней стабильной версии программного обеспечения;
- личные телефоны пользователей приложения должны быть защищены программно-аппаратными комплексами контроля и управления доступом;
- личные телефоны Администратора должны быть защищены надежным паролем, состоящим более чем из 8 символов (строчные буквы, прописные буквы, знаки препинания и цифры);
- личные телефоны пользователей должны быть защищены надежным паролем, состоящим более чем из 8 символов (строчные буквы, прописные буквы, знаки препинания и цифры);
- помещение, в котором устанавливается аппаратная часть серверной инфраструктуры, должно быть оборудовано системой контроля и управления доступом.

Заключение

В результате выполнения выпускной квалификационной работы было проведено исследование предметной области, Приложение функциональной диагностики оказывает помощь пациентам, изучая наиболее важные основные этапы оказания помощи пациенту, диагностируя его заболевание и собирая его данные для получения правильного результата. Это начальная версия программы, также поможем пациенту добраться до врача для консультации и предложим больницы и лекарства для типа его заболевания, а также ближайшие к нему аптеки.

В данной ВКР были поэтапно решены следующие задачи:

- произведено проектирование структуры, в которой подробно рассмотрены основные концепции и принципы развития применения определения заболеваний с помощью ИИ и ряд других функций;
- проведено сравнение приложений, где были учтены все характеристики и преимущества;
- исследовались новые фреймворки, различные алгоритмы и библиотеки. Более подробно изучен алгоритм линейной регрессии, принцип работы и обучения этого алгоритма. Также была реализована общая структура для разработки алгоритма линейной регрессии и его рабочее объяснение;
- мобильное приложение для функциональной диагностики, и тоже нахождение ближайших больниц и аптек. Также имеются дополнительные справочные системы.

Список использованных источников

1. Артюнина, Г.П. Основы медицинских знаний: Здоровье, болезнь и образ жизни: учебное пособие для высшей школы / Г.П. Артюнина, С.А. Игнаткова. – 2-е изд., перераб. – М.: Академический Проект, 2004. – 560 с.
2. Бароненко, В.А., Рапопорт, Л.А. Здоровье и физическая культура студента / Под ред. В.А. Бароненко: Учеб. пособие. М.: Альфа-М, 2003. – 352 с.
3. Основы медицинских знаний: учеб. пособие / Р.И. Айзман, В.Г. Бубнов, В.Б. Рубанович, М.А. Суботялов. – Новосибирск: АРТА, 2011. – 224 с.
4. Зайцев, А.Г. Педагогика счастья / Г.К. Зайцев, А.Г. Зайцев. – СПб.: Издательство «Союз», 2002. – 320 с.
5. Давиденко Д.Н., Щедрин Ю.Н., Щеголев В.А. Здоровье и образ жизни студентов / Под. общ. ред. проф. Д.Н. Давиденко: Учебное пособие. – СПб.: СПбГУИТМО, 2005. – 124 с.
6. Тимушкина Н.В., Талагаева Ю.А. Здоровый образ жизни / Под. общ. ред. Тимушкина Н.В., Талагаева Ю.А.: Учебное пособие. – Саратов: Саратовский источник, 2005. – 124 с.
7. Хэлзи // HELZY.RU: сервис определения возможных заболеваний по симптомам. 2022. URL: <http://www.helzy.ru> (дата обращения: 09.04.2022).
8. Машинное обучение // MACHINE LEARNING: профессиональный информационно-аналитический ресурс, посвященный машинному обучению, распознаванию образов и интеллектуальному анализу данных. 2022. URL: <http://www.machinelearning.ru> (дата обращения: 25.04.2022).
9. Sharden B., Massaron L., Boschetti A. "Large-scale machine learning with Python" DMK Press Publishing House, 2018 - 358 p.

Приложение А

Листинг программы

```
package com.example.symptomchecker;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.NavUtils;
import android.app.AlertDialog;
import android.app.LoaderManager;
import android.content.ContentValues;
import android.content.CursorLoader;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.Loader;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.Menu;
import android.view.MenuItem;
import android.view.MotionEvent;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
import com.example.symptomchecker.data.IllnessContract.IllnessEntry;

public class IllnessEditorActivity extends AppCompatActivity implements
LoaderManager.LoaderCallbacks<Cursor> {

    /** Identifier for the doctor data loader */
    private static final int EXISTING_ILLNESS_LOADER = 0;

    /** Content URI for the existing doctor (null if it's a new doctor) */
    private Uri mCurrentIllnessUri;

    private EditText nameTextView;
    private EditText typeTextView;
    private EditText descTextView;
    private EditText causesTextView;
    private EditText riskTextView;
    private EditText symptomsTextView;
    private EditText medicinesTextView;
    private EditText askTextView;

    private boolean mIllnesHasChanged = false;

    private View.OnTouchListener mTouchListener = new View.OnTouchListener() {
        @Override
        public boolean onTouch(View view, MotionEvent motionEvent) {
            mIllnesHasChanged = true;
            return false;
        }
    }
```

```

};

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_illness_editor);

    Intent intent = getIntent();
    mCurrentIllnessUri = intent.getData();

    if (mCurrentIllnessUri == null) {
        setTitle(getString(R.string.editor_activity_title_new_doctor));
        invalidateOptionsMenu();
    } else {
        // Initialize a loader to read the doctor data from the database
        // and display the current values in the editor
        setTitle(getString(R.string.editor_activity_title_edit_doctor));
        getLoaderManager().initLoader(EXISTING_ILLNESS_LOADER, null, this);
    }

    // Find all relevant views that we will need to read user input from
    nameTextView = (EditText) findViewById(R.id.edit_illness_name);
    typeTextView = (EditText) findViewById(R.id.edit_illness_type);
    descTextView = (EditText) findViewById(R.id.edit_illness_desc);
    causesTextView = (EditText) findViewById(R.id.edit_illness_causes);
    riskTextView = (EditText) findViewById(R.id.edit_illness_risk);
    symptomsTextView = (EditText) findViewById(R.id.edit_illness_symptoms);
    medicinesTextView = (EditText) findViewById(R.id.edit_illness_medicines);
    askTextView = (EditText) findViewById(R.id.edit_illness_ask);

    // Setup OnTouchListeners on all the input fields, so we can determine if the user
    // has touched or modified them. This will let us know if there are unsaved changes
    // or not, if the user tries to leave the editor without saving.
    nameTextView.setOnTouchListener(mTouchListener);
    typeTextView.setOnTouchListener(mTouchListener);
    descTextView.setOnTouchListener(mTouchListener);
    causesTextView.setOnTouchListener(mTouchListener);
    riskTextView.setOnTouchListener(mTouchListener);
    symptomsTextView.setOnTouchListener(mTouchListener);
    medicinesTextView.setOnTouchListener(mTouchListener);
    askTextView.setOnTouchListener(mTouchListener);
}

@Override
public void onBackPressed() {
    // If the pet hasn't changed, continue with handling back button press
    if (!mIllnesHasChanged) {
        super.onBackPressed();
        return;
    }
}

```

```

// Otherwise if there are unsaved changes, setup a dialog to warn the user.
// Create a click listener to handle the user confirming that changes should be discarded.
DialogInterface.OnClickListener discardButtonClickListener =
    new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialogInterface, int i) {
            // User clicked "Discard" button, close the current activity.
            finish();
        }
    };

// Show dialog that there are unsaved changes
showUnsavedChangesDialog(discardButtonClickListener);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu options from the res/menu/menu_editor.xml file.
    // This adds menu items to the app bar.
    getMenuInflater().inflate(R.menu.menu_doctor_editor, menu);
    return true;
}

@Override
public boolean onPrepareOptionsMenu(Menu menu) {
    super.onPrepareOptionsMenu(menu);
    // If this is a new pet, hide the "Delete" menu item.
    if (mCurrentIllnessUri == null) {
        MenuItem menuItem = menu.findItem(R.id.action_delete);
        menuItem.setVisible(false);
    }
    return true;
}

@Override
public Loader<Cursor> onCreateLoader(int i, Bundle bundle) {
    // Since the editor shows all pet attributes, define a projection that contains
    // all columns from the hosp table
    String[] projection = {
        IllnessEntry._ID,
        IllnessEntry.COLUMN_ILLNESS_NAME,
        IllnessEntry.COLUMN_ILLNESS_TYPE,
        IllnessEntry.COLUMN_ILLNESS_DESC,
        IllnessEntry.COLUMN_ILLNESS_CAUSES,
        IllnessEntry.COLUMN_ILLNESS_RISK,
        IllnessEntry.COLUMN_ILLNESS_SYMPTOMS,
        IllnessEntry.COLUMN_ILLNESS_MEDICINES,
        IllnessEntry.COLUMN_ILLNESS_ASKDOCTOR,
    };
}

```

```

// This loader will execute the ContentProvider's query method on a background thread
return new CursorLoader(this, // Parent activity context
    mCurrentIllnessUri, // Provider content URI to query
    projection, // Columns to include in the resulting Cursor
    null, // No selection clause
    null, // No selection arguments
    null);
}

@Override
public void onLoadFinished(Loader<Cursor> loader, Cursor cursor) {
    if (cursor == null || cursor.getCount() < 1) {
        return;
    }

    if (cursor.moveToFirst()) {

        // Find the columns of hosp attributes that we're interested in
        int ColumnIndex1 =
cursor.getColumnIndex(IllnessEntry.COLUMN_ILLNESS_NAME);
        int ColumnIndex2 = cursor.getColumnIndex(IllnessEntry.COLUMN_ILLNESS_TYPE);
        int ColumnIndex3 = cursor.getColumnIndex(IllnessEntry.COLUMN_ILLNESS_DESC);
        int ColumnIndex4 =
cursor.getColumnIndex(IllnessEntry.COLUMN_ILLNESS_CAUSES);
        int ColumnIndex5 = cursor.getColumnIndex(IllnessEntry.COLUMN_ILLNESS_RISK);
        int ColumnIndex6 =
cursor.getColumnIndex(IllnessEntry.COLUMN_ILLNESS_SYMPTOMS);
        int ColumnIndex7 =
cursor.getColumnIndex(IllnessEntry.COLUMN_ILLNESS_MEDICINES);
        int ColumnIndex8 =
cursor.getColumnIndex(IllnessEntry.COLUMN_ILLNESS_ASKDOCTOR);

        // Read the hosp attributes from the Cursor for the current hos
        String a1 = cursor.getString(ColumnIndex1);
        String a2 = cursor.getString(ColumnIndex2);
        String a3 = cursor.getString(ColumnIndex3);
        String a4 = cursor.getString(ColumnIndex4);
        String a5 = cursor.getString(ColumnIndex5);
        String a6 = cursor.getString(ColumnIndex6);
        String a7 = cursor.getString(ColumnIndex7);
        String a8 = cursor.getString(ColumnIndex8);

        nameTextView.setText(a1);
        typeTextView.setText(a2);
        descTextView.setText(a3);
        causesTextView.setText(a4);
        riskTextView.setText(a5);
        symptomsTextView.setText(a6);
        medicinesTextView.setText(a7);
        askTextView.setText(a7);

    }
}

```

```

}

@Override
public void onLoaderReset(Loader<Cursor> loader) {
    nameTextView.setText("");
    typeTextView.setText("");
    descTextView.setText("");
    causesTextView.setText("");
    riskTextView.setText("");
    symptomsTextView.setText("");
    medicinesTextView.setText("");
    askTextView.setText("");
}

public boolean onOptionsItemSelected(MenuItem item) {
    // User clicked on a menu option in the app bar overflow menu
    switch (item.getItemId()) {
        // Respond to a click on the "Save" menu option
        case R.id.action_save:
            // Save pet to database
            saveIllness();
            Intent intent = new Intent(IllnessEditorActivity.this, IllnessActivity.class);
            startActivity(intent);
            return true;
        // Respond to a click on the "Delete" menu option
        case R.id.action_delete:
            // Pop up confirmation dialog for deletion
            showDeleteConfirmationDialog();
            return true;
        // Respond to a click on the "Up" arrow button in the app bar
        case android.R.id.home:
            // If the pet hasn't changed, continue with navigating up to parent activity
            // which is the {@link CatalogActivity}.
            if (!mIllnesHasChanged) {
                NavUtils.navigateUpFromSameTask(IllnessEditorActivity.this);
                return true;
            }

            // Otherwise if there are unsaved changes, setup a dialog to warn the user.
            // Create a click listener to handle the user confirming that
            // changes should be discarded.
            DialogInterface.OnClickListener discardButtonClickListener =
                new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialogInterface, int i) {
                        // User clicked "Discard" button, navigate to parent activity.
                        NavUtils.navigateUpFromSameTask(IllnessEditorActivity.this);
                    }
                };

            // Show a dialog that notifies the user they have unsaved changes
            showUnsavedChangesDialog(discardButtonClickListener);

```

```

        return true;
    }
    return super.onOptionsItemSelected(item);
}

// Save and delete
private void saveIllness() {
    // Read from input fields
    // Use trim to eliminate leading or trailing white space
    String nameString = nameTextView.getText().toString().trim();
    String a2 = typeTextView.getText().toString().trim();
    String a3 = descTextView.getText().toString().trim();
    String a4 = causesTextView.getText().toString().trim();
    String a5 = riskTextView.getText().toString().trim();
    String a6 = symptomsTextView.getText().toString().trim();
    String a7 = medicinesTextView.getText().toString().trim();
    String a8 = askTextView.getText().toString().trim();

    // Check if this is supposed to be a new pet
    // and check if all the fields in the editor are blank
    if (mCurrentIllnessUri == null &&
        TextUtils.isEmpty(nameString) &&
        TextUtils.isEmpty(a2) && TextUtils.isEmpty(a3) && TextUtils.isEmpty(a4) &&
        TextUtils.isEmpty(a5) &&
        TextUtils.isEmpty(a6) && TextUtils.isEmpty(a7) ) {
        // Since no fields were modified, we can return early without creating a new doctor.
        // No need to create ContentValues and no need to do any ContentProvider operations.
        return;
    }

    // Create a ContentValues object where column names are the keys,
    // and pet attributes from the editor are the values.
    ContentValues values = new ContentValues();
    values.put(IllnessEntry.COLUMN_ILLNESS_NAME, nameString);
    values.put(IllnessEntry.COLUMN_ILLNESS_TYPE, a2);
    values.put(IllnessEntry.COLUMN_ILLNESS_DESC, a3);
    values.put(IllnessEntry.COLUMN_ILLNESS_CAUSES, a4);
    values.put(IllnessEntry.COLUMN_ILLNESS_RISK, a5);
    values.put(IllnessEntry.COLUMN_ILLNESS_SYMPTOMS, a6);
    values.put(IllnessEntry.COLUMN_ILLNESS_MEDICINES, a7);
    values.put(IllnessEntry.COLUMN_ILLNESS_ASKDOCTOR, a8);

    // Determine if this is a new or existing pet by checking if mCurrentPetUri is null or not
    if (mCurrentIllnessUri == null) {
        // This is a NEW pet, so insert a new pet into the provider,
        // returning the content URI for the new pet.
        Uri newUri = getContentResolver().insert(IllnessEntry.CONTENT_URI, values);

        // Show a toast message depending on whether or not the insertion was successful.
        if (newUri == null) {
            // If the new content URI is null, then there was an error with insertion.

```

```

        Toast.makeText(this, getString(R.string.editor_insert_doctor_failed),
            Toast.LENGTH_SHORT).show();
    } else {
        // Otherwise, the insertion was successful and we can display a toast.
        Toast.makeText(this, getString(R.string.editor_insert_doctor_successful),
            Toast.LENGTH_SHORT).show();
    }
} else {
    // Otherwise this is an EXISTING pet, so update the pet with content URI:
mCurrentPetUri
    // and pass in the new ContentValues. Pass in null for the selection and selection args
    // because mCurrentPetUri will already identify the correct row in the database that
    // we want to modify.
    int rowsAffected = getContentResolver().update(mCurrentIllnessUri, values, null, null);

    // Show a toast message depending on whether or not the update was successful.
    if (rowsAffected == 0) {
        // If no rows were affected, then there was an error with the update.
        Toast.makeText(this, getString(R.string.editor_update_doctor_failed),
            Toast.LENGTH_SHORT).show();
    } else {
        // Otherwise, the update was successful and we can display a toast.
        Toast.makeText(this, getString(R.string.editor_update_doctor_successful),
            Toast.LENGTH_SHORT).show();
    }
}
}

private void deleteIllnes() {
    // Only perform the delete if this is an existing pet.
    if (mCurrentIllnessUri != null) {
        // Call the ContentResolver to delete the pet at the given content URI.
        // Pass in null for the selection and selection args because the mCurrentPetUri
        // content URI already identifies the pet that we want.
        int rowsDeleted = getContentResolver().delete(mCurrentIllnessUri, null, null);

        // Show a toast message depending on whether or not the delete was successful.
        if (rowsDeleted == 0) {
            // If no rows were deleted, then there was an error with the delete.
            Toast.makeText(this, getString(R.string.editor_delete_doctor_failed),
                Toast.LENGTH_SHORT).show();
        } else {
            // Otherwise, the delete was successful and we can display a toast.
            Toast.makeText(this, getString(R.string.editor_delete_doctor_successful),
                Toast.LENGTH_SHORT).show();
        }
    }
}

// Close the activity
finish();
}

```

```

    /* dialog */
    private void showUnsavedChangesDialog(DialogInterface.OnClickListener
discardButtonClickListener) {
        // Create an AlertDialog.Builder and set the message, and click listeners
        // for the postivie and negative buttons on the dialog.
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setMessage(R.string.unsaved_changes_dialog_msg);
        builder.setPositiveButton(R.string.discard, discardButtonClickListener);
        builder.setNegativeButton(R.string.keep_editing, new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                // User clicked the "Keep editing" button, so dismiss the dialog
                // and continue editing the pet.
                if (dialog != null) {
                    dialog.dismiss();
                }
            }
        });

        // Create and show the AlertDialog
        AlertDialog alertDialog = builder.create();
        alertDialog.show();
    }

    /**
     * Prompt the user to confirm that they want to delete this pet.
     */
    private void showDeleteConfirmationDialog() {
        // Create an AlertDialog.Builder and set the message, and click listeners
        // for the postivie and negative buttons on the dialog.
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setMessage(R.string.delete_dialog_msg);
        builder.setPositiveButton(R.string.delete, new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                // User clicked the "Delete" button, so delete the pet.
                deleteIllnes();
            }
        });
        builder.setNegativeButton(R.string.cancel, new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                if (dialog != null) {
                    dialog.dismiss();
                }
            }
        });

        // Create and show the AlertDialog
        AlertDialog alertDialog = builder.create();
        alertDialog.show();
    }
}

```