**Содержание**

[Введение 2](#_Toc163403492)

[1. Аналитическая часть 4](#_Toc163403493)

[1.1. Интервью с заказчиком 4](#_Toc163403494)

[1.2. Описание предметной области 5](#_Toc163403495)

[1.3. Описание вариантов использования приложения 7](#_Toc163403496)

[1.4. Классификация пользователей результатов проекта 8](#_Toc163403497)

[1.5. Обоснование выбора средств разработки 8](#_Toc163403498)

[1.6. Описание структуры данных 10](#_Toc163403499)

[1.7. Описание архитектуры приложения 10](#_Toc163403500)

[1.8. Проектирование интерфейса 12](#_Toc163403501)

[2. Технологическая часть 16](#_Toc163403502)

[2.1 Организация данных 16](#_Toc163403503)

[2.2 Разработка дизайна приложения – разметка + листнинг 17](#_Toc163403504)

[2.3 Разработка API 21](#_Toc163403505)

[2.4 Протоколы тестирования 22](#_Toc163403506)

[2.5 Разграничение прав доступа 24](#_Toc163403507)

[3 Экономическая часть 25](#_Toc163403508)

[3.1 Описание качества программного продукта 25](#_Toc163403509)

[3.2 Управление процессом разработки 26](#_Toc163403510)

[3.3 Предварительная оценка трудоемкости разработки 27](#_Toc163403511)

[Заключение 29](#_Toc163403512)

[Использованные источники информации 30](#_Toc163403513)

## Введение

Медицина занимает важную роль в жизни человека. Во многих медицинских учреждениях, главной задачей является обеспечение быстрого обслуживания пациентов.

Актуальность создания мобильного приложения для записи к врачу заключается в необходимости использования постоянно увеличивающегося объема информации при решении различных медицинских задач.

Целью курсового проекта является создание мобильного приложения, которое позволит улучшить процесс работы медицинских учреждений, а также обслуживание пациентов.

Объектом разработки является процесс обслуживания населения в медицинской сфере.

Предметом исследования является система для автоматизации процесса обслуживания пациентов.

Реализация выше поставленной цели будет возможна после решения ряда задач:

1. Проектирование и разработка базы данных для хранения информации о врачах, пациентах и записях на прием.

2. Реализовать пользовательский интерфейс для поиска врачей;

3. Реализовать возможность просмотра расписания и записи на прием;

Функциональные требования, выявленные при разработке мобильного приложения:

1. Регистрация и авторизация пользователя;

2. Возможность присваивания роли пользователю;

3. Возможность поиска врачей;

4. Возможность записи на прием;

5. Возможность добавления услуги и свободного времени для приема.

В дальнейшем разработанное приложение может быть внедрено в медицинское учреждение, клиники и медицинские центры, а также использовано пациентами для записи на прием к врачу в различных сферах медицины.

Основанием для разработки проекта является учебный план специальности 09.02.07 «Информационные системы и программирование».

## 1. Аналитическая часть

## 1.1. Интервью с заказчиком

Интервью с заказчиком, это процесс беседы с представителем заказчика или ответственным за проект лицом, в результате которого собирается информация о требованиях, целях, ожиданиях заказчика необходимое для разработки и совершенствования продукта. Вот несколько вопросов, которые могут быть заданы в интервью с заказчиком:

1. Что является главной причиной для создания приложения для записи к врачу?

Мы хотим улучшить процесс работы медицинских учреждений, а также обслуживание пациентов.

1. Какие основные функции и возможности вы хотели бы видеть в приложении?

Во-первых, нам нужна функция поиска врачей, чтобы пациенты могли найти нужного специалиста по имени, специализации. Во-вторых, нужна возможность просмотра информации о каждом враче. И возможность записаться на прием у выбранного врача.

1. Какую информацию вы хотели бы получать о пользователях и записях на прием?

Понадобится такая информация как: имя, фамилия, адрес электронной почты, номер телефона.

1. Какое взаимодействие должно быть между приложением и вашей базой данных о врачах и пациентах?

Нам нужно, чтобы приложение было интегрировано с нашей существующей базой данных, чтобы пациенты могли видеть актуальную информацию о врачах.

1. Важны ли для вас какие-либо дополнительные функции или требования?

Нам важна защита данных пациентов и их конфиденциальность.

## 1.2. Описание предметной области

Предметная область приложения "Запись к врачу" связана с облегчением процесса записи пациента на консультацию или прием к врачу в медицинских учреждениях. Она охватывает широкий спектр деятельности, связанный с организацией и обеспечением комфортного взаимодействия между пациентами и врачами.

В предметной области приложения учитываются различные аспекты, такие как доступность информации о врачах и их расписаниях, возможность выбора удобного времени приема, регистрация новых пациентов и обновление существующих данных, отмене или изменении времени.

В целом, предметная область приложения "Запись к врачу" связана с улучшением организации и доступности медицинской помощи, повышением удобства для пациентов и эффективностью работы врачей. Она включает в себя основные этапы процесса записи к врачу, начиная от поиска врача и выбора удобного времени, заканчивая получением квалифицированной медицинской помощи и оценкой качества услуг.

Также данное мобильное приложение имеет функции регистрации/авторизации с выбором роли (пациент или врач).

Исходя из выявленных функциональных требований была построена Use-case диаграмма, представленная на рисунке 1:

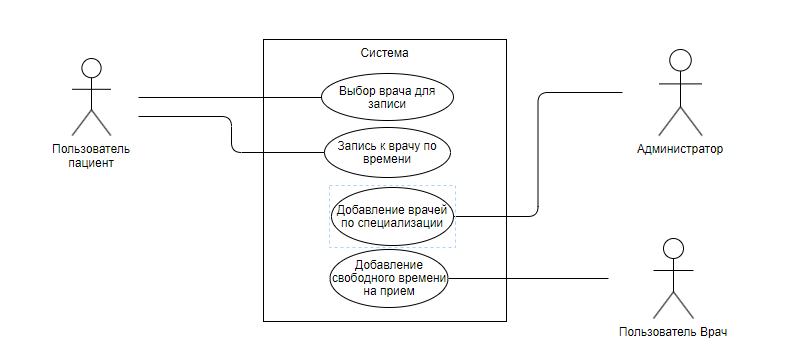


Рисунок 1. Use-case диаграмма.

На основе составленной Use-case диаграммы было выделено 4 прецедента. Рассмотрим прецедент “Выбор врача для записи”:

Таблица 1 — Прецедент “Выбор врача для записи”

|  |  |
| --- | --- |
| Название прецедента | Выбор врача для записи. |
| Цель | Запись на прием к врачу. |
| Актеры | Пользователь пациент, Пользователь Врач, Администратор, Система. |
| Ссылки (предусловия) | Реализованы прецеденты: Авторизация пациента в системе. |
| Результаты (постусловия) | Пользователь пациент выбирает интересующего врача, записывается к врачу на свободное время. |

## 1.3. Описание вариантов использования приложения

На основе выявленных прецедентов в Use-case диаграмме был составлен основной успешный сценарий, представленный в таблице 2:

Таблица 2 — Основной вариант использования для пациента.

|  |  |
| --- | --- |
| Действия актера | Отклик системы |
| 2. Пользователь пациент выбирает  врача для записи из списка.  4. Пользователь пациент записывается к врачу на свободное время. | 1. Система выдала список врачей. 2. Система отображает свободное время   для записи пользователю.   1. Система отображает время записи и ФИО пациента врачу, а пациенту ФИО врача, кабинет, время. |

На основе успешного варианта использования можно составить альтернативный поток событий:

1. Произошла ошибка в системе. Система выдает сообщение об ошибке.

2. Данные отсутствуют в системе. Система выдает сообщение об ошибке.

Таблица 3 — Основной вариант использования для врача.

|  |  |
| --- | --- |
| Действия актера | Отклик системы |
| 1. Пользователь врач авторизуется в системе.  2. Пользователь врач просматривает данные пациентов.  4. Пользователь врач меняет данные на прием. | 3. Система выдает информацию о пациентах.  5. Система записывает ФИО пациента и резервирует время для приема. |

На основе успешного варианта использования можно составить альтернативный поток событий:

1. Произошла ошибка в системе. Система выдает сообщение об ошибке.

2. Данные отсутствуют в системе. Система выдает сообщение об ошибке.

## 1.4. Классификация пользователей результатов проекта

Классификация пользователей результатов проекта для приложения запись к врачу может быть следующей:

1. Администраторы/врачи: Эта группа пользователей, которые создают профили врачей в приложении. Они предоставляют информацию о себе, своей квалификации, специализации, доступных услугах и расписании приемов. Могут просматривать и управлять своим расписанием, принимать/отклонять запросы на прием. Они имеют специальные права доступа для администрирования и управления системой.

2. Пациенты: В эту группу входят сами пациенты, которые создают учетные записи в приложении для получения доступа к его функциональным возможностям. Эти пользователи могут просматривать профили врачей, расписание приемов, совершать запись на прием, отменять записи, получать уведомления и управлять своими личными данными.

# 1.5. Обоснование выбора средств разработки

Выбор средства разработки Kotlin может быть обоснован несколькими основаниями:

1. Интероперабельность с Java: Kotlin является полностью совместимым с Java, что означает, что код на Java может без проблем использоваться в проекте на Kotlin и наоборот. Это позволяет разработчикам постепенно переводить существующий код с Java на Kotlin, не требуя полной переписывания.

2. Безопасность: Kotlin предоставляет некоторые дополнительные механизмы безопасности по сравнению с Java. Например, Kotlin предотвращает ошибки присваивания значения null, которые являются одной из наиболее распространенных ошибок программирования.

3. Удобство и простота: Код на Kotlin обычно более краток и выразителен по сравнению с Java, что делает его более читабельным и легко поддерживаемым. Кotlin также имеет множество удобных функций, таких как функции-расширения и соглашения, которые упрощают работу с языком.

4. Поддержка инструментария и сообщества: Kotlin имеет широкую поддержку различных инструментов разработки, таких как Android Studio, IntelliJ IDEA и другие популярные IDE. Кроме того, Kotlin имеет активное разработчиками сообщество, готовое помочь и поддержать новых пользователей.

5. Повышение производительности: Благодаря своим синтаксическим особенностям и дополнительным возможностям, Kotlin может ускорить процесс разработки и снизить количество ошибок. Это может привести к более эффективной работе команды разработчиков и улучшению производительности проекта в целом.

Несомненно, выбор средства разработки является субъективным и может зависеть от конкретных потребностей и предпочтений команды разработчиков. Однако, с учетом удобства, безопасности, интероперабельности и активного сообщества, Kotlin является привлекательным выбором для разработки приложений для записи к врачу.

# 1.6. Описание структуры данных

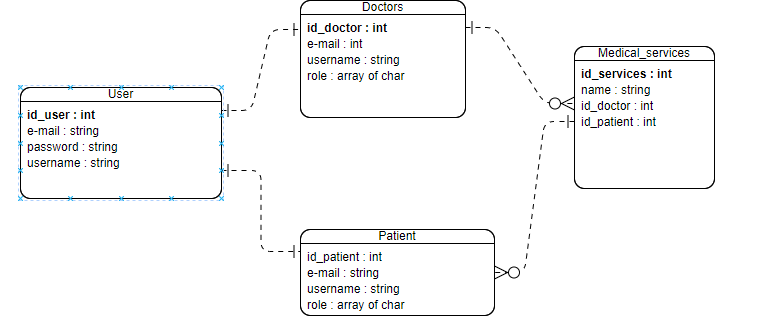
По результатам описания предметной области была составлена ERD диаграмма, представленная на рисунке 2.

Рисунок 2. ER – Диаграмма

# 1.7. Описание архитектуры приложения

Архитектура приложения для записи к врачу может быть организована в соответствии с принципами модульности и разделения ответственности. Вот возможное описание основных компонентов архитектуры приложения:

1. Клиентский интерфейс: Это часть приложения, которая взаимодействует с пользователями - пациентами. Она включает в себя пользовательский интерфейс (UI), который позволяет пациентам авторизоваться в приложении, выбрать нужного врача и записаться на прием.

2. Бизнес-логика: Этот компонент отвечает за обработку и управление данными, логикой бизнес-процессов и выполнением операций. Он включает в себя функции для создания, редактирования и удаления записей на прием, а также обработки записей на прием.

3. Серверная инфраструктура: Эта часть архитектуры отвечает за хранение данных приложения и взаимодействие с базой данных. Она включает в себя серверное API, которое обрабатывает запросы от клиента, а также модули доступа к данным, которые обеспечивают взаимодействие с базой данных.

4. База данных: В этой части архитектуры хранятся все данные, связанные с пациентами и врачами. База данных должна быть спроектирована таким образом, чтобы обеспечить эффективное хранение и извлечение данных, а также обеспечить целостность данных с помощью соответствующих ограничений и отношений.

5. Аутентификация и безопасность: Это компоненты, отвечающие за аутентификацию и авторизацию пациентов, а также за защиту данных приложения. Безопасность может включать в себя шифрование данных, управление правами доступа и механизмы защиты от несанкционированного доступа.

При проектировании приложения была приведена следующая архитектура “Клиент-сервер”, представленная на рисунке 3

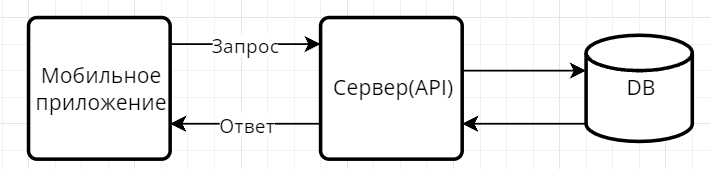


Рисунок 3. Схема архитектуры приложения

# 1.8. Проектирование интерфейса

При анализе предметной области был спроектирован проект интерфейса приложения, представленный на рисунках 4-9. Каждая форма выполняет определенную функцию.

|  |  |
| --- | --- |
| Рисунок 4. Форма “Авторизация” | Рисунок 5. Форма “Регистрация” |

|  |  |
| --- | --- |
| Рисунок 6. Окно “Выбор врача” | Рисунок 7. Окно “Запись к врачу” |

|  |  |
| --- | --- |
| Рисунок 8. Окно “Добавление услуги и свободного времени для приема” | Рисунок 9. Окно “Список записанных” |

После создания проекта интерфейса была создана схема перемещений по приложению, представленная на рисунке 10:

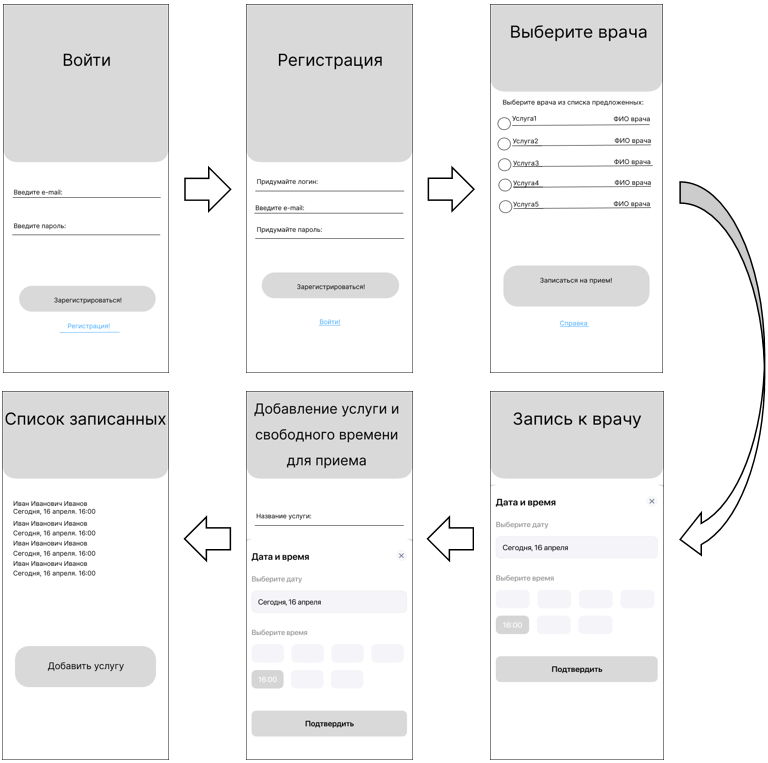


Рисунок 10. Окно “ Схема перемещений по приложению”

На рисунке ясно видно, как происходит работа в приложении.

# 2. Технологическая часть

## 2.1 Организация данных

Организация данных в приложении для записи к врачу играет важную роль в эффективной работе приложения. На следующем рисунке представлена схема базы данных:

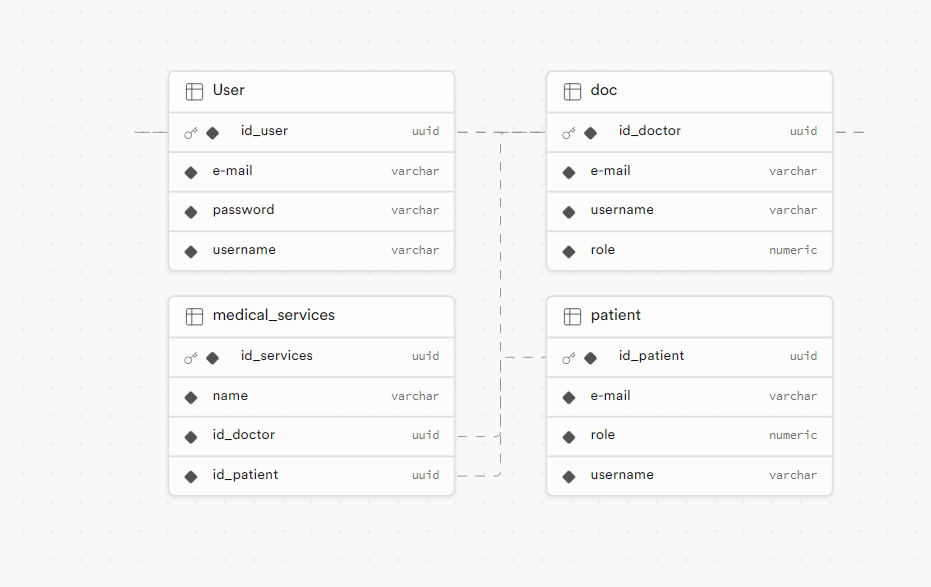


Рисунок 11. Схема базы данных

2.2 Разработка дизайна приложения – разметка + листнинг

Разработка интерфейса мобильного приложения для записи к врачу требует внимательного планирования и проектирования. При разработке будет реализовано шесть окон интерфейса представленные на рисунках 12-15.

****

Рисунок 12. Окно авторизации пользователя.

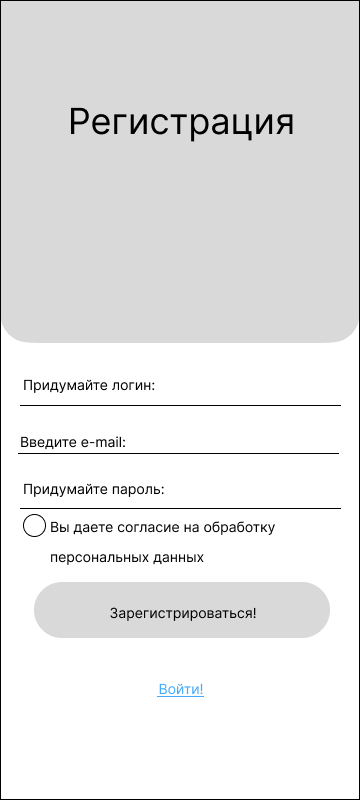


Рисунок 13. Окно регистрации пользователя.

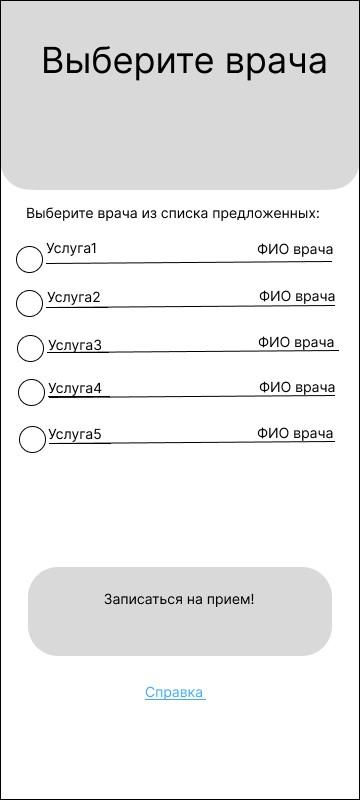


Рисунок 14. Окно выбора врача.

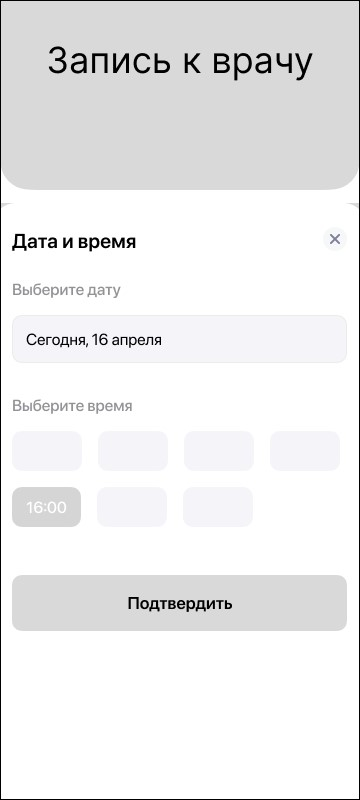


Рисунок 15. Окно выбора даты и времени для записи к врачу.

2.3 Разработка API

При разработке API для мобильного приложения “Запись к врачу” будет использована оболочка базы данных Supabase основанная на PostgreSQL. PostgreSQL является надежной и расширяемой реляционной базой данных, которая обеспечивает множество возможностей и функций для работы с данными. Имплементация библиотеки Supabase, представлена в листинг 1:

Листинг 1 – имплементирование библиотеки Supabase:

implementation(platform("io.github.jan-tennert.supabase:bom:VERSION"))

implementation("io.github.jan-tennert.supabase:postgrest-kt")

Для реализации функциональных требований были реализованы запросы: регистрация, авторизация, извлечения данных, вставки данных, обновление данных. Данные запросы представленные в листингах 2-5:

Листинг 2 запрос регистрации:

val user = supabase.auth.signUpWith(Email) {

email = email\_text

password = password\_text

}

Листинг 3 запрос авторизации:

supabase.auth.signInWith(Email) {

email = email\_text

password = password\_text

}

Листинг 4 запрос авторизации:

supabase.auth.signInWith(Email) {

email = email\_text

password = password\_text

}

Листинг 5 запрос извлечение данных:

val city = supabase.from("user").select().decodeSingle<user>()

Листинг 6 запрос вставки данных:

val test = test(name = "test1", number tasks = RecyclerItem)

supabase.from("cities").insert(city)

2.4 Протоколы тестирования

Протоколы тестирования – это документы, которые определяют последовательность и спецификацию шагов, которые должны быть выполнены для проведения тестов или проверки определенной функциональности. Для тестирования мобильного приложения мы будем использовать Тест-кейс, представленный на таблицах 4-5.

Таблица 4. Тест-кейс регистрация в приложении

|  |  |  |
| --- | --- | --- |
| Название: | Тест регистрации | |
| Предусловие | Ожидаемый результат | Результат теста |
| Открытие приложения на странице регистрации. | Мобильное приложения для записи к врачу открыто и доступно. |  |
| Шаги теста: | | |
| Заполните форму регистрации в приложении:  “Логин”: Ввести “Test”  “e-mail”: Ввести “Test”  “Пароль”: Ввести “test” | Данные успешно введены, и приложение перешло на страницу выбора услуг. | Данные успешно введены, что подтверждается переходом на страницу выбора услуг. |

Таблица 5. Тест-кейс авторизации в приложении

|  |  |  |
| --- | --- | --- |
| Название: | Тест регистрации | |
| Предусловие | Ожидаемый результат | Результат теста |
| Открытие приложения на странице авторизации. | Мобильное приложения для записи к врачу открыто и доступно. |  |
| Шаги теста: | | |
| Заполните форму авторизации в приложении:  “e-mail”: Ввести “Test”  “Пароль”: Ввести “test” | Данные успешно введены, и приложение перешло на страницу выбора услуг. | Данные успешно введены, что подтверждается переходом на страницу выбора услуг. |

## 2.5 Разграничение прав доступа

Разграничение прав доступа является важной составляющей системы записи к врачу, так как позволяет обеспечить безопасность данных пациентов и контроль доступа к функционалу приложения. Можно предложить следующую систему разграничения прав доступа:  
1. Пользователь:  
- Может просматривать список доступных врачей и их профили с основной информацией.  
- Может просматривать расписание врачей и доступные слоты для записи.  
- Может создавать учетную запись или входить в существующую.  
- Может записываться на прием к врачу.  
- Может просматривать свои собственные записи на прием и их статус.  
- Может отменять свои записи на прием.  
2. Врач:  
- Может просматривать свое личное расписание и занятые слоты времени.  
- Может просматривать информацию о своих пациентах и состоянии их записей на прием.  
- Может подтверждать или отклонять записи на прием от пациентов.  
- Может добавлять информацию о своей недоступности для записи на прием.  
- Может внести изменения в информацию о своем профиле (контактные данные, расписание и т. д.).  
3. Администратор:  
- Может управлять списком доступных врачей в системе (добавление, редактирование, удаление).  
- Может просматривать и управлять информацией о пациентах (например, редактирование контактных данных).  
- Может просматривать и редактировать информацию о записях на прием.  
- Может управлять учетными записями пользователей (создание, блокировка, удаление).

# Экономическая часть

## 3.1 Описание качества программного продукта

Данный программный продукт разработан на основе созданной базы данных.

Все функции отвечают требованиям поставленной задачи, протестированы и не имеют ошибок.

**Функциональные возможности**

Программный продукт соответствует требованиям поставленной задачи, протестирован и не имеет ошибок.

Только авторизованные пользователи могут вносить изменения.

**Надежность**

Средства проверки достоверности исходных данных - вывод производится из базы данных, элементы таблиц которой были загружены ранее.

**Практичность**

Данный проект является продуктом закрытого программного кода, поэтому нет возможности у любого человека редактировать данный программный продукт.

**Сопровождаемость**

Функции не конфликтуют друг с другом, в связи с этим ошибок при работе с программой не возникает.

**3.2 Управление процессом разработки**

Перед проведением оценки трудоемкости выполнения проекта проведена декомпозиция работ, выполняемых для достижения целей проекта. Проведен анализ обобщенной операционной структуры трудозатрат для программных проектов, расписание проекта не потребовало дополнений.

На рисунках 16, 17 представлена иерархическая структура работ на диаграмме Ганта.



Рисунок 16 – иерархическая структура работ.

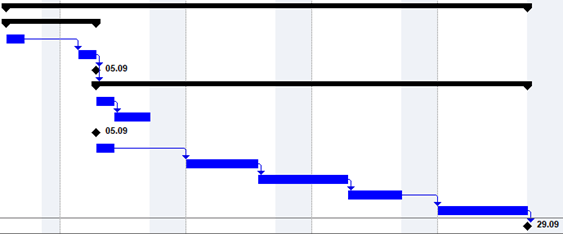


Рисунок 17 – Диаграмма Ганта.

## 3.3 Предварительная оценка трудоемкости разработки

На основании перечня функций, выявленных на этапе анализа требований была выполнена экспертная оценка трудоемкости разработки приложения, с учетом его функций, представленная в таблице 3.

Таблица 3 – Экспертная оценка трудоемкости разработки приложения

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Функция** | **Лучший случай** | **Наиболее вероятный** | **Худший случай** | **Ожидаемый случай** |
| Добавление врача по специальности. | 4 | 6 | 26 | 8 |
| Запись к врачу по времени. | 2 | 4 | 20 | 6,3 |
| Выбор врача по записи. | 2 | 4 | 20 | 6,3 |
| Добавление свободного времени на прием. | 2 | 4 | 20 | 6,3 |
| **Итого** | **10** | **18** | **86** | **26,9** |

В предварительной оценке трудоёмкости разработки представлено количество дней, за которое возможно реализовать функции из Use-Case диаграммы используя формулу PERT (табл.3).

Оценку средней трудоемкости по каждому элементарному пакету можно определить по формуле:

(1)

1. Добавление врача по специальности.

Oi = 4

Mi = 6

Pi = 26

Средняя трудоемкость рассчитана по формуле (1):

1. Запись к врачу по времени.

Oi = 2

Mi = 4

Pi = 20

Средняя трудоемкость рассчитана по формуле (1):

1. Выбор врача по записи.

Oi = 2

Mi = 4

Pi = 20

Средняя трудоемкость рассчитана по формуле (1):

1. Добавление свободного времени на прием.

Oi = 2

Mi = 4

Pi = 20

Средняя трудоемкость рассчитана по формуле (1):

На основе экспертная оценки трудоемкости разработки мобильного приложения было выявлено, что ожидаемый случай выполнения разработки программного продукта займет 26,9 дней.

# Заключение

В современном мире с развитием технологий и мобильных приложений стало возможным улучшить доступность и удобство получения медицинской помощи. Реализация приложения для записи к врачу позволяет пациентам сэкономить время, получить информацию о доступности врачей и услуг, записаться на прием, а также взаимодействовать с медицинским персоналом через электронную систему.

В результате проведенного исследования и разработки приложения, было создано удобное и интуитивно понятное решение, которое позволяет пациентам с легкостью выбирать врачей по их специализации, просматривать расписание приемов и занимать свободные слоты. Приложение обеспечивает безопасность и конфиденциальность информации, а также предоставляет возможности для обратной связи и получения уведомлений о приеме и изменениях в расписании.

Кроме того, приложение способствует оптимизации работы врачей и администраторов, позволяя автоматизировать процесс записи и управления расписанием, уменьшая вероятность ошибок и улучшая эффективность работы медицинского учреждения.

В целом, приложение для записи к врачу представляет огромный потенциал для улучшения качества медицинской помощи и удовлетворения потребностей пациентов. Это современный, инновационный и эффективный инструмент, который может помочь сократить время ожидания, повысить доступность и повысить уровень обслуживания в медицинских учреждениях.

# Использованные источники информации

1. Солдатенкова Е. С. (2015). Управление долгосрочным развитием информационных систем в медицине. Издательство "Питер".
2. Киселев Ю. (2016). Новые требования к разработке медицинского ПО. Системные аспекты информационных технологий, 26(1), 36-42.
3. Яшкин В. (2017). Использование информационных технологий в управлении медицинскими организациями. IT Manager, 7, 63-66
4. Алиева Е. (2018). Опыт применения мобильного приложения для записи к врачу. Здравоохранение, 9, 63-66.
5. Малашенко М. (2019). Оптимизация процесса записи на прием к врачу с помощью приложений на платформе iOS. Управление качеством и нормативно-правовое регулирование здравоохранения, 1, 61-65
6. Новикова Д. (2020). Как развить мобильное приложение для записи к врачу. Медицинские информационные технологии, 9, 26-29.
7. Смирнова О. (2021). Возможности мобильных приложений в организации работы медицинских учреждений. Здравоохранение, 10, 52-55.
8. Левшина Е., Леонтьева О. (2021). Преимущества использования электронной системы записи на прием к врачу. Здравоохранение, 2, 44-47.
9. Заяц А. М. Введение в гибридные технологии разработки мобильных приложений : учебное пособие для СПО / А. М. Заяц, Н. П. Васильева. – 2-е изд., стер. – Санкт-Петербург : Лань, 2022. – 160 с.;
10. Васильев Н. П. Введение в гибридные технологии разработки мобильных приложений : учебное пособие для вузов / Н. П. Васильев, А. М. Заяц. – 3-е изд., стер. – Санкт-Петербург : Лань, 2022. – 160 с.;
11. Старолетов С. М. Основы тестирования программного обеспечения : учебное пособие для СПО / С. М. Старолетов. – 3-е изд., стер. – Санкт-Петербург : Лань, 2023. – 192 с.;
12. Старолетов С. М. Основы тестирования и верификации программного обеспечения : учебное пособие для вузов / С. М. Старолетов. – 3-е изд., стер. – Санкт-Петербург : Лань, 2023. – 344 с.;
13. ГОСТ Р 58412-2019 «Защита информации»;
14. ГОСТ Р ИСО/МЭК 25010-2015 «Системная и программная инженерия»;
15. ГОСТ 34.602-2020 «Комплекс стандартов на автоматизированные системы».