Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Модели данных и системы управления базами данных

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему

**ПРОГРАММНОЕ СРЕДСТВО ДЛЯ ОРГАНИЗАЦИИ РАБОТЫ МЕДИЦИНСКОГО ЦЕНТРА**

БГУИР КП 1-40 04 01 011 ПЗ

|  |  |  |
| --- | --- | --- |
| Студент |  | Д. А. Демидова |
| Руководитель |  | А. В. Давыдчик |
|  |  |  |

Минск 2024

СОДЕРЖАНИЕ

[Введение 4](#__RefHeading___Toc12078_602624272)

[1 Архитектура вычислительной системы 5](#__RefHeading___Toc11247_602624272)

[1.1 Структура и архитектура вычислительной системы 5](#__RefHeading___Toc11249_602624272)

[1.2 История, версии и достоинства 7](#__RefHeading___Toc11251_602624272)

[1.3 Обоснование выбора вычислительной системы 9](#__RefHeading___Toc11253_602624272)

[2 Платформа программного обеспечения 11](#__RefHeading___Toc11259_602624272)

[2.1 Выбор операционной системы 11](#__RefHeading___Toc11261_602624272)

[2.2 Выбор платформы для написания программы 12](#__RefHeading___Toc11263_602624272)

[3 Теоретическое обоснование разработки программного продукта 13](#__RefHeading___Toc11265_602624272)

[3.1 Обоснование необходимости разработки 13](#__RefHeading___Toc11267_602624272)

[3.2 Технологии программирования, используемые для решения](#__RefHeading___Toc11269_602624272) [поставленных задач 13](#__RefHeading___Toc11271_602624272)

[4 Проектирование функциональных](#__RefHeading___Toc11277_602624272) [возможностей программы 14](#__RefHeading___Toc11279_602624272)

[4.1 Подключение к базе данных 14](#__RefHeading___Toc11281_602624272)

[4.2 Регистрация и авторизация пользователей 14](#__RefHeading___Toc11283_602624272)

[4.3 Управление пользователями 14](#__RefHeading___Toc11285_602624272)

[4.4 Взаимодействие с сущностями приложения 14](#__RefHeading___Toc11287_602624272)

[4.5 Общее описание системы 14](#__RefHeading___Toc11289_602624272)

[4.6 Руководство пользователя 14](#__RefHeading___Toc11291_602624272)

5 Проектирование разрабатываемой базы данных программного   
обеспечения 15

[5.1 Разработка информационной модели 15](#__RefHeading___Toc11281_6026242721)

[5.2 ER-диаграмма базы данных 15](#__RefHeading___Toc11283_6026242721)

[5.3 Оптимизация структуры разработанной базы данных 15](#__RefHeading___Toc11285_6026242721)

[5.4 Описание базы данных 15](#__RefHeading___Toc11287_6026242721)

[Заключение 16](#__RefHeading___Toc11307_602624272)

[Список литературных источников 17](#__RefHeading___Toc11309_602624272)

[Приложение А](#__RefHeading___Toc11311_602624272) [(обязательное)](#__RefHeading___Toc11313_602624272) [Листинг программного кода 18](#__RefHeading___Toc11315_602624272)

[Приложение Б](#__RefHeading___Toc11317_602624272) [(обязательное)](#__RefHeading___Toc11319_602624272) [Конечная схема базы данных 19](#__RefHeading___Toc11321_602624272)

[Приложение В](#__RefHeading___Toc11333_602624272) [(обязательное)](#__RefHeading___Toc11335_602624272) [Ведомость курсового проекта 20](#__RefHeading___Toc11337_602624272)

# **ВВЕДЕНИЕ**

Современные медицинские учреждения ежедневно сталкиваются с необходимостью обработки большого объема данных, включая персональные данные пациентов, истории болезни, расписания врачей и другую информацию. Применение базы данных для хранения этих данных позволяет централизовать управление и организовать эффективное взаимодействие между сотрудниками медицинского центра. Правильно спроектированная система также позволяет соблюдать требования безопасности и конфиденциальности, которые являются важными аспектами работы с персональной информацией в медицинской сфере.

Целью данной курсовой работы является создание программного средства для организации работы медицинского центра, которое будет включать базу данных для хранения и управления информацией о пациентах, сотрудниках, услугах и медицинских записях. Создаваемое программное средство должно облегчить администрирование и ускорить доступ к медицинской информации, что особенно важно для обеспечения качественного и оперативного обслуживания пациентов.

Исходя из цели проекта был составлен следующий перечень задач:

1 Определить и обосновать перечень информационных сущностей для выбранной предметной области.

2 Разработать структуру базы данных, которая охватывает все необходимые аспекты работы медицинского центра, такие как управление пациентами, персоналом и медицинскими услугами.

3 Реализовать механизм взаимодействия с сущностями приложения.

4 Создать приложение, использующее разработанную базу данных.

5 Создать графический интерфейс для взаимодействия с приложением, использующим разработанную базу данных.

В ходе разработки программного средства будут использованы принципы проектирования и современные технологии для создания надежной и безопасной базы данных, которая станет важным инструментом для работы медицинского центра. В конечном итоге данная система должна упростить рабочие процессы и минимизировать ошибки, связанные с человеческим фактором, что будет способствовать повышению качества обслуживания пациентов.

# 1 АРХИТЕКТУРА ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ

## 1.1 Структура и архитектура вычислительной системы

PostgreSQL – это объектно-реляционная система управления базами данных (ORDBMS), наиболее развитая из открытых СУБД в мире. Имеет открытый исходный код и является альтернативой коммерческим базам данных. С помощью PostgreSQL можно создавать, хранить базы данных и работать с данными с помощью запросов на языке SQL.

Одной из наиболее сильных сторон СУБД PostgreSQL является архитектура. Как и в случаях со многими коммерческими СУБД, PostgreSQL можно применять в среде клиент-сервер – это предоставляет множество преимуществ и пользователям, и разработчикам.

В основе PostgreSQL – серверный процесс базы данных, выполняемый на одном сервере. Доступ из приложений к данным базы PostgreSQL производится с помощью специального процесса базы данных. То есть клиентские программы не могут получать самостоятельный доступ к данным даже в том случае, если они функционируют на том же ПК, на котором осуществляется серверный процесс.пользователям, и разработчикам.

В основе PostgreSQL – серверный процесс базы данных, выполняемый на одном сервере. Доступ из приложений к данным базы PostgreSQL производится с помощью специального процесса базы данных. То есть клиентские программы не могут получать самостоятельный доступ к данным даже в том случае, если они функционируют на том же ПК, на котором осуществляется серверный процесс. [ <https://otus.ru/nest/post/1584/> ]

Типичная модель распределенного приложения СУБД PostgreSQL (рисунок 1.1):

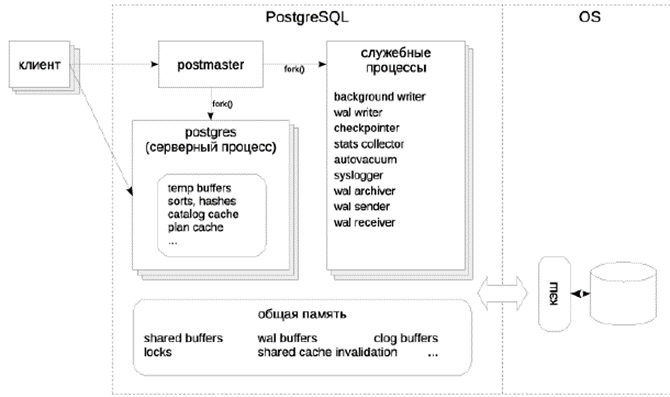


Рисунок 1.1 – Схема СУБД PostgreSQL

СУБД PostgreSQL ориентирована на протокол TCP/IP (локальная сеть либо Интернет), при этом каждый клиент соединён с главным серверным процессом БД (на рисунке 1.1 этот процесс обозначен Postmaster). Именно Postmaster создает новый серверный процесс специально в целях обслуживания запросов на доступ к данным определенного клиента. [ <https://habr.com/ru/companies/otus/articles/706346/> ]

Сервер PostgreSQL может обрабатывать несколько одновременных подключений от клиентов. Для этого он запускает новый процесс для каждого соединения. С этого момента клиент и новый серверный процесс обмениваются данными без вмешательства исходного процесса postgres. Таким образом, процесс сервера-супервизора всегда работает, ожидая клиентских подключений, в то время как клиентские и связанные серверные процессы приходят и уходят. [ <https://otus.ru/nest/post/1584/> ]

Данные, которыми управляет PostgreSQL, хранятся в базах данных. Один экземпляр PostgreSQL одновременно работает с несколькими базами, которые вместе называются кластером баз данных.

Каталог, в котором размещаются все файлы, относящиеся к кластеру, обычно называют словом PGDATA, по имени переменной окружения, указывающей на этот каталог.

При инициализации в PGDATA создаются три одинаковые базы данных (рисунок 1.2):

1 template0 используется, например, для восстановления из логической резервной копии или для создания базы в другой кодировке и никогда не должна меняться;

2 template1 служит шаблоном для всех остальных баз данных, которые может создать пользователь в этом кластере;

3 postgres представляет собой обычную базу данных, которую можно использовать по своему усмотрению.

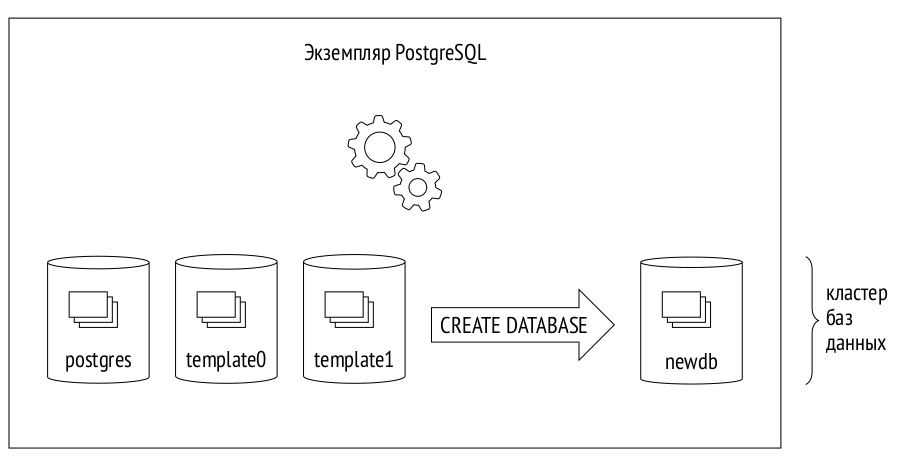


Рисунок 1.2 – Кластер PostgreSQL

Метаинформация обо всех объектах кластера (таких как таблицы, индексы, типы данных или функции) хранится в таблицах, относящихся к системному каталогу. В каждой базе данных имеется собственный набор таблиц (и представлений), описывающих объекты этой конкретной базы. Существует также несколько таблиц системного каталога, общих для всего кластера, которые не принадлежат какой-либо определенной базе данных и доступны в любой из них. [ книга postgresql изнутри ]

## 1.2 История, версии и достоинства

Ранние версии системы были основаны на старой программе POSTGRES University, созданной университетом Беркли: так появилось название PostgreSQL. И сейчас СУБД иногда называют «Постгрес». Существуют сокращения PSQL и PgSQL – они тоже обозначают PostgreSQL.

По состоянию на июнь 2024 года PostgreSQL занимает четвертое место в общемировом рейтинге популярных СУБД (рисунок 1.1). [ <https://www.statista.com/statistics/809750/worldwide-popularity-ranking-database-management-systems/> ]

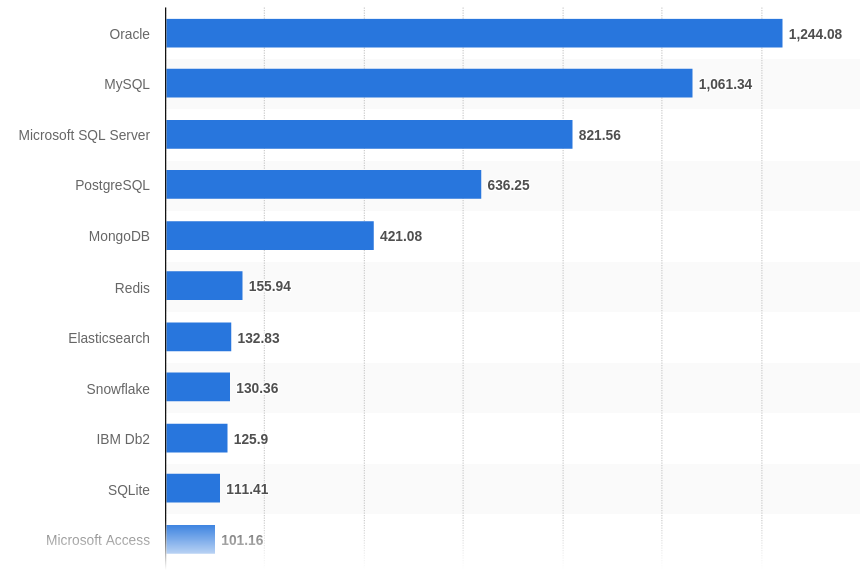


Рисунок 1.1 – Рейтинг популярности СУБД в июне 2024 года

У СУБД PostgreSQL много преимуществ, которые продолжают повышать ее популярность:

1 Любой специалист может бесплатно скачать, установить СУБД и сразу начать работу с базами данных.

2 PostgreSQL подходит для работы в любой операционной системе: Linux, macOS, Windows. Пользователь получает систему «из коробки» – чтобы установить и использовать программу, не нужны дополнительные инструменты.

3 PostgreSQL поддерживает много разных типов и структур данных, в том числе сетевые адреса, данные в текстовом формате JSON и геометрические данные для координат геопозиций. Все эти форматы можно хранить и обрабатывать в СУБД. Также при работе с PostgreSQL можно создавать собственные типы данных, их называют пользовательскими.

4 Размер базы данных в PostgreSQL не ограничен и зависит от того, сколько свободной памяти есть в месте хранения: на сервере, локальном компьютере или в облаке.

5 PostgreSQL реализует принципы ACID. Это четыре требования для надежной работы систем, которые обрабатывают данные в режиме реального времени. Если все требования выполняются, данные не будут теряться из-за технических ошибок или сбоев в работе оборудования.

6 PostgreSQL поддерживает все современные функции баз данных: оконные функции, вложенные транзакции, триггеры.

7 Хотя большинство операций в PostgreSQL и используют классический стандарт языка SQL, помимо него поддерживается и свой отдельный диалект, позволяющий еще комфортнее писать запросы.

8 Поддерживается репликация «из коробки». Репликация – это сохранение копии базы данных. Копия может находиться на другом сервере.

9 PostgreSQL позволяет быстро без потерь перенести данные из другой СУБД. [ <https://practicum.yandex.ru/blog/chto-takoe-subd-postgresql/> ]

10 Возможность одновременного доступа к базе с нескольких устройств. В СУБД реализована клиент-серверная архитектура, когда база данных хранится на сервере, а доступ к ней осуществляется с клиентских компьютеров. Для ситуаций, когда несколько человек одновременно модифицируют базу используется технология MVCC – Multiversion Concurrency Control, многоверсионное управление параллельным доступом.

Благодаря перечисленным выше преимуществам иногда PostgreSQL называют бесплатным аналогом Oracle Database. Обе системы адаптированы под большие проекты и высокую нагрузку. Но есть разница: они по-разному хранят данные, предоставляют разные инструменты и различаются возможностями. Важная особенность PostgreSQL в том, что эта система – feature-rich: так называют проекты с широким функционалом. [ <https://blog.skillfactory.ru/glossary/postgresql/> ]

## 1.3 Обоснование выбора вычислительной системы

PostgreSQL выбрана для разработки приложения для медицинского центра, поскольку она сочетает в себе мощные функциональные возможности, надежность и высокую популярность среди реляционных СУБД. К июню 2024 года PostgreSQL занимает четвертое место в мировом рейтинге популярных СУБД, что подтверждает ее востребованность в сфере информационных технологий.

Одно из ключевых преимуществ PostgreSQL – это свободная лицензия, которая позволяет загружать и использовать систему без затрат. Она поддерживается на всех популярных операционных системах, включая Linux, macOS и Windows, что делает ее доступной для самых разных приложений и сред. Система предоставляет «из коробки» все необходимые инструменты, позволяя сразу приступить к работе с базами данных, не требуя дополнительных программ или надстроек.

PostgreSQL выделяется поддержкой различных типов и структур данных, что особенно важно для приложений, которые обрабатывают сложные данные. В дополнение к стандартным типам данных она позволяет работать с JSON, геометрическими данными и сетевыми адресами. В данном проекте это облегчает хранение медицинской информации и расширяет возможности ее обработки.

Другой важный аспект – масштабируемость PostgreSQL. Размер базы данных ограничен лишь объемом доступной памяти на сервере или в облаке, что позволяет хранить большие объемы медицинских данных и обеспечивает длительное использование системы без необходимости миграции.

Надежность системы обусловлена ее соответствием стандартам ACID, что важно для приложений, которые требуют целостности и безопасности данных. Медицинские данные должны быть защищены от потерь и технических сбоев, и PostgreSQL предлагает стабильную платформу для их обработки и хранения.

Также PostgreSQL реализует многоверсионное управление параллельным доступом (MVCC), что позволяет нескольким пользователям одновременно работать с базой данных без блокировки доступа. Это ключевое преимущество для медицинского центра, где несколько сотрудников могут одновременно обращаться к базе для записи, просмотра и обновления данных пациентов.

Таким образом, PostgreSQL была выбрана для этого проекта благодаря ее широким функциональным возможностям, высокой надежности, поддержке разнообразных типов данных и оптимальной масштабируемости. Это делает PostgreSQL идеальной СУБД для медицинского центра, нуждающегося в эффективной и надежной системе управления данными.

# 2 ПЛАТФОРМА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## 2.1 Выбор операционной системы

Linux – оптимальный выбор для приложения медицинского центра, поскольку данная операционная система обеспечивает надежную, безопасную и гибкую платформу для серверных решений, особенно тех, которые требуют высокой производительности и стабильности.

Linux известен своей высокой стабильностью, особенно на серверных платформах. В медицинском центре работа с данными пациентов и организация процесса требует круглосуточной доступности и минимального времени простоя. Linux стабильно работает даже при значительных нагрузках, что делает его идеальным для приложений с высоким объемом запросов и большим количеством данных.

Медицинские данные требуют особого уровня защиты, так как они содержат персональные данные и конфиденциальную информацию. Сильная встроенная система прав доступа (основанная на ролях) и возможность регулярных обновлений безопасности делают Linux безопасной ОС для хранения и управления конфиденциальными данными.

Также Linux позволяет оптимально использовать память и процессорные мощности, что особенно важно для приложений, обрабатывающих большое количество запросов и данных, как это требуется в медицинском центре. Благодаря тому, что Linux занимает минимум системных ресурсов, оставляя больше мощности для самого приложения и базы данных PostgreSQL, приложение будет работать быстрее и устойчивее.

Поскольку Linux является системой с открытым исходным кодом, она предоставляет экономически выгодное решение для медицинского центра. Установка и использование Linux не требуют затрат на лицензии, что снижает затраты на внедрение и эксплуатацию.

Кроме того, Linux предлагает множество инструментов для автоматизации задач: скрипты на Bash, планировщики задач (например, cron) и возможности контейнеризации (например, Docker). Эти инструменты можно использовать для регулярного резервного копирования базы данных, обновлений системы и настройки мониторинга. Возможность автоматизации поможет медицинскому центру обеспечить бесперебойную работу и уменьшить затраты на обслуживание системы.

Благодаря этим преимуществам, Linux является надежной и эффективной платформой для приложений медицинского центра, требующих высокой безопасности, производительности и стабильности.

Таким образом, в качестве операционной системы для проведения сравнения используется Linux (дистрибутив Ubuntu).

## 2.2 Выбор платформы для написания программы

В качестве языка программирования для написания программы используется Python. Python имеет несколько преимуществ для разработки приложения медицинского центра:

1 Python отличается простой и понятной синтаксической структурой, что сокращает время на написание и поддержку кода. Это позволяет разработчику быстрее перейти от идеи к работающему продукту, а также легко поддерживать и обновлять приложение.

2 Python предлагает широкий выбор библиотек и фреймворков для разработки серверных приложений, таких как Django и FastAPI. Эти фреймворки упрощают настройку и структуру серверной части, обеспечивая готовые решения для обработки запросов, аутентификации и работы с базами данных. Например, Django ORM позволяет упростить взаимодействие с PostgreSQL, делая его интуитивно понятным и надежным.

3 Python имеет обширные возможности для работы с базами данных. Библиотеки, такие как psycopg2 и SQLAlchemy, облегчают взаимодействие с PostgreSQL, позволяя выполнять как простые, так и сложные запросы к базе данных. Эти библиотеки предоставляют как интерфейс ORM, так и позволяют отправлять базы чистые запросы через так называемый «raw SQL».

В качестве платформы для разработки был выбран PyCharm. Главная причина этому то, что PyCharm является одной из наиболее популярных и мощных IDE для разработки на Python, предоставляя разработчикам множество инструментов и удобств для повышения производительности и качества кода, и в том числе для удобной интеграции с базой данных и отслеживания ее состояния.

# 3 ТЕОРЕТИЧЕСКОЕ ОБОСНОВАНИЕ РАЗРАБОТКИ ПРОГРАММНОГО ПРОДУКТА

## 3.1 Обоснование необходимости разработки

Эффективное управление данными является ключевым фактором успешной работы медицинских учреждений. Медицинские центры ежедневно сталкиваются с необходимостью обработки больших объемов информации: от записи пациентов на прием и хранения медицинской истории до планирования расписания врачей и управления финансовыми операциями. Без надлежащей системы автоматизации такие задачи требуют значительных временных и человеческих ресурсов, что может приводить к ошибкам, задержкам и снижению уровня обслуживания пациентов.

Создание программного средства для организации работы медицинского центра позволяет оптимизировать эти процессы и повысить общую эффективность работы учреждения. Программное обеспечение, разработанное на основе базы данных, дает возможность централизованного хранения, обработки и анализа данных. Это способствует более быстрому доступу к информации, снижает риск потери данных и минимизирует человеческий фактор при выполнении рутинных операций.

Кроме того, внедрение автоматизированной системы упрощает выполнение многих задач, включая управление расписанием врачей, обработку заявок от пациентов, формирование отчетов для руководства и мониторинг занятости ресурсов. Такие функции особенно актуальны в условиях высокой нагрузки, когда ручное управление данными становится непрактичным.

Необходимость разработки подобного программного обеспечения обусловлена и современными требованиями в области медицины, включая соблюдение стандартов безопасности данных пациентов. Программное средство, основанное на современных СУБД, таких как PostgreSQL, позволяет обеспечить высокий уровень надежности, целостности и защиты данных.

Важно отметить, что разработка такого программного средства не только упрощает управление данными, но и способствует улучшению качества предоставляемых услуг. Пациенты получают возможность оперативного взаимодействия с медицинским центром, будь то запись на прием или получение медицинской документации. Руководство же получает мощный инструмент для анализа ключевых показателей работы учреждения и принятия обоснованных решений.

Таким образом, разработка программного средства для организации работы медицинского центра является важным шагом к повышению эффективности управления, улучшению качества обслуживания пациентов и внедрению современных технологий в медицинскую сферу. Такое решение отвечает как текущим потребностям медицинских центров, так и тенденциям их цифровой трансформации.

## 3.2 Технологии программирования, используемые для решения

## поставленных задач

Для решения задач курсового проекта был сделан выбор в пользу минималистичного набора технологий. Основной акцент был сделан на глубокой интеграции с базой данных, чтобы максимально эффективно управлять данными, сохраняя контроль за выполнением операций на уровне SQL-запросов.

Одной из ключевых библиотек, использованных в разработке, является psycopg2 – официальная библиотека для работы с PostgreSQL в Python. Она предоставляет удобный и гибкий интерфейс для выполнения SQL-запросов, управления транзакциями, а также работы с курсорами. Такой подход к реализации позволяет разработчику напрямую взаимодействовать с базой данных, избегая промежуточных абстракций, которые могли бы усложнить контроль за процессом выполнения операций. Например, все транзакции в программе (коммиты и откаты) управляются вручную, что гарантирует предсказуемое и безопасное выполнение операций с данными. [ <https://www.psycopg.org/docs/> ]

Для создания графического пользовательского интерфейса было выбрано средство tkinter. Это встроенная в Python библиотека, которая позволяет разрабатывать удобные и функциональные интерфейсы без необходимости использования сторонних инструментов. Ее использование обеспечивает простоту интеграции с основной логикой программы и позволяет создать интерфейс, полностью адаптированный под поставленные задачи. [ <https://docs.python.org/3/library/tk.html> ]

Дополнительно, для упрощения работы с базой данных в процессе разработки был использован инструмент Docker. Это решение позволило избежать необходимости установки PostgreSQL на локальной машине разработчика, что делает разработку более гибкой и удобной. Использование Docker обеспечивает возможность быстрого развертывания базы данных в изолированном контейнере, позволяя минимизировать риски конфликтов зависимостей и упрощая настройку среды разработки. Более того, данный подход легко масштабируется и обеспечивает удобство переноса проекта на другие машины или серверы, что делает его особенно актуальным для проектов с использованием реляционных баз данных. [ <https://docs.docker.com/> ]

Несмотря на относительно небольшой набор используемых библиотек и инструментов, это решение было осознанным и связано с целью сосредоточить внимание на работе с базой данных через сырой SQL. Такой подход позволяет полностью контролировать процесс взаимодействия с данными, учитывать особенности выполнения транзакций и корректно обрабатывать возможные ошибки. В отличие от ORM (Object-Relational Mapping), данный подход исключает дополнительные абстракции и дает гибкость в использовании возможностей PostgreSQL. [ <https://medium.com/@ritika.adequate/the-pros-and-cons-of-using-raw-sql-versus-orm-for-database-development-e9edde7ee31e> ]

Таким образом, выбранный стек технологий соответствует задачам курсовой работы, обеспечивая не только необходимую функциональность, но и высокую степень контроля за всеми этапами работы с базой данных. На основе этого можно сделать вывод о том, что используемые технологии программирования позволяют выполнить цели данного курсового проекта.

# 4 ПРОЕКТИРОВАНИЕ ФУНКЦИОНАЛЬНЫХ

# ВОЗМОЖНОСТЕЙ ПРОГРАММЫ

## 4.1 Функциональные требования

Программа должна иметь систему ролей, состоящую из администратора, доктора и пациента, так как их пользовательский опыт в рамках медицинского центра отличается.

Администратор должен быть способен осуществлять CRUD-операции над всеми сущностями программы, иметь доступ ко всем данным.

Пациент, чтобы успешно пользоваться врачебными услугами, должен иметь возможность:

1 Просмотреть список своих записей к докторам;

2 Записаться к доктору на свободное окошко и выбрать предоставляемую услугу;

3 Просмотреть список всех когда-либо выписанных ему диагнозов (аналогично медицинской книжке в поликлинике);

4 Просмотреть список всех когда-либо выданных предписаний от докторов.

Доктор, чтобы успешно оказывать врачебные услуги, должен иметь возможность:

1 Просмотреть список записанных к нему пациентов;

2 Просматривать детальную информацию о посещении, такую как тип оказываемой услуги, дату и время приема;

3 По итогам проведенного приема добавлять в медицинскую книжку пациента какой-либо диагноз, а также врачебное предписание;

4 Помечать прием, как закрытый.

Кроме того, система конечно же должна предоставлять пользователям возможность зарегистрироваться, заполнить свой профиль, а также заходить в приложение по логину и паролю.

## 4.2 Подключение к базе данных

Для более экономного расходования ресурсов компьютера база данных поднимается не локально, а в Docker. Это не только избавляет пользователей от необходимости тратить время на установку базы данных локально и ее конфигурацию, но также занимает гораздо меньше памяти, так как был подобран Docker-образ (Docker-image), имеющий оптимальный баланс между производительностью и размером – Postgres 16 в версии Alpine, который весит всего 275 мегабайт. Запускается такая база данных одной простой командой «docker compose up».

На уровне приложения подключение к базе данных осуществляется с помощью библиотеки psycopg2. В приложении есть отдельный класс для работы с базой данных Database, который хранит подключение к базе в одном из своих полей. Именно этот класс используется во всех контроллерах приложения, что позволяет разделить код, связанный с взаимодействием с базой данных, и код, обрабатывающий действия пользователя. Кроме того, если база данных еще не настроена (например в случае, если это первый запуск приложения), то данный класс вызовет функции create\_database и create\_tables, которые запустят нужные SQL-скрипты для настройки базы данных.

Также данный класс в стиле raw-SQL выполняет все запросы к базе данных, а также контролирует фиксацию (commit) и откат (rollback) изменений.

## 4.3 Регистрация и авторизация пользователей

To be continued

## 4.4 Функционал администратора

To be continued

## 4.5 Функционал доктора

To be continued

## 4.6 Функционал пациента

To be continued

Таким образом, … .

**5 ПРОЕКТИРОВАНИЕ РАЗРАБАТЫВАЕМОЙ БАЗЫ ДАННЫХ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ**

To be continued

## 5.1 Разработка концептуальной модели

To be continued

## 5.2 Разработка логической модели

To be continued

## 5.3 Разработка физической модели

To be continued

## 5.4 Конечная модель базы данных

To be continued

# ЗАКЛЮЧЕНИЕ

В рамках данного курсового проекта … .

Таким образом, … .

# СПИСОК ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ

[1] Архитектура Zen: сколько поколений продержится главная технология AMD [Электронный ресурс]. – Режим доступа: https://club.dns-shop.ru/blog/t-100-protsessoryi/61416-arhitektura-zen-skolko-pokolenii-proderjitsya-glavnaya-tehnologi/ – Дата доступа: 01.10.2023.

[2] Поколения процессоров AMD Ryzen [Электронный ресурс]. – Режим доступа: https://te4h.ru/pokoleniya-protsessorov-amd-ryzen – Дата доступа: 01.10.2023.

# ПРИЛОЖЕНИЕ А

# (обязательное)

# Листинг программного кода

To be continued

# ПРИЛОЖЕНИЕ Б

# (обязательное)

# Конечная схема базы данных

# ПРИЛОЖЕНИЕ В

# (обязательное)

# Ведомость курсового проекта