

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

Кафедра _____ «Прикладная математика и информатика» _____
(наименование)

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

Разработка программного обеспечения
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка программного обеспечения информационной системы
для анализа медицинских данных»

Обучающийся _____ А.Ю. Портнов _____
(Инициалы Фамилия) (личная подпись)

Руководитель _____ к.т.н., Н.В. Хрипунов _____
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2024

Аннотация

Тема бакалаврской работ «Разработка программного обеспечения информационной системы для анализа медицинских данных».

Ключевые слова: разработка программного обеспечения, информационно-аналитическая система, анализ медицинских данных.

Объектом исследования бакалаврской работы является информационная система для анализа медицинских данных.

Предметом исследования является программное обеспечение информационной системы для анализа медицинских данных.

Цель выпускной квалификационной работы – разработка программного обеспечения информационной системы для анализа медицинских данных.

Практическая значимость бакалаврской работы заключается в разработке ПО информационной системы для анализа медицинских данных, обеспечивающего высокую эффективность проведенного анализа.

Данная работа состоит из введения, трех глав, заключения и списка используемой литературы и источников.

Первая глава посвящена формированию требований и постановке задачи на разработку ПО информационной системы для анализа медицинских данных. Вторая глава посвящена проектированию ПО информационной системы для анализа медицинских данных. В третьей главе описан процесс реализации и тестирования ПО информационной системы для анализа медицинских данных.

В заключении описываются результаты выполнения выпускной квалификационной работы.

Бакалаврская работа состоит из 45 страниц текста, 24 рисунков, 9 таблиц и 24 источников.

Оглавление

Введение.....	5
Глава 1 Постановка задачи на разработку программного обеспечения информационной системы для анализа медицинских данных.....	7
1.1 Программное обеспечение медицинской информационно-аналитической системы.....	7
1.2 Формирование требований на разработку программного обеспечения информационной системы для анализа медицинских данных.....	8
1.3 Обзор и анализ аналогов программного обеспечения информационной системы для анализа медицинских данных	10
1.3.1 Платформа для анализа данных RapidMiner.....	10
1.3.2 Аналитическая платформа Logitom	11
1.3.3 Аналитическая платформа KNIME.....	12
Глава 2 Проектирование программного обеспечения информационной системы для анализа медицинских данных.....	16
2.1 Выбор методологии проектирования программного обеспечения информационной системы для анализа медицинских данных	16
2.2 Логическое моделирование программного обеспечения информационной системы для анализа медицинских данных	20
2.2.1 Разработка функциональной модели ПО.	22
2.2.2 Разработка архитектуры ПО.....	25
Глава 3 Реализация и тестирование программного обеспечения информационной системы для анализа медицинских данных.....	28
3.1 Выбор средства разработки программного обеспечения информационной системы для анализа медицинских данных	28
3.2 Реализация программного обеспечения информационной системы для анализа медицинских данных	31
3.3 Тестирование программного обеспечения информационной системы для анализа медицинских данных	35

Заключение	41
Список используемой литературы и используемых источников.....	43

Введение

Анализ данных в здравоохранении жизненно важен. Он помогает организациям здравоохранения оценивать и развивать практикующих врачей, выявлять аномалии при сканировании и прогнозировать вспышки заболеваний.

Анализ данных также может снизить затраты организаций здравоохранения и повысить эффективность бизнес-аналитики. Самое главное, что он помогает медицинскому персоналу клиник ставить правильные диагнозы и принимать более эффективные решения по уходу за пациентами.

Для анализа медицинских данных используются разные аналитические инструменты. По своим функциональным и архитектурным особенностям эти инструменты относятся к категории информационно-аналитических систем.

Как показывает практика, качество анализа и эффективность полученных результатов зависят от программного обеспечения (ПО), которое реализует используемые в информационной системе алгоритмы анализа данных.

В этой связи представляет научно-практический интерес разработка программного обеспечения информационной системы для анализа медицинских данных, обеспечивающего высокую эффективность проведенного анализа.

Объектом исследования бакалаврской работы является информационная система для анализа медицинских данных.

Предметом исследования является программное обеспечение информационной системы для анализа медицинских данных.

Цель выпускной квалификационной работы – разработка программного обеспечения информационной системы для анализа медицинских данных.

Для достижения данной цели необходимо решить следующие задачи:

– сформулировать требования и произвести постановку задачи на

разработку программного обеспечения информационной системы для анализа медицинских данных;

- выполнить проектирование программного обеспечения информационной системы для анализа медицинских данных;
- реализовать программное обеспечение для автоматизации тестирования бизнес-приложений и провести его тестирование.

Методы исследования – методы анализа медицинских данных, методы и технологии разработки ПО.

Практическая значимость бакалаврской работы заключается в разработке ПО информационной системы для анализа медицинских данных, обеспечивающего высокую эффективность проведенного анализа.

Данная работа состоит из введения, трех глав, заключения и списка используемой литературы и источников.

Первая глава посвящена формированию требований и постановке задачи на разработку ПО информационной системы для анализа медицинских данных.

Вторая глава посвящена проектированию ПО информационной системы для анализа медицинских данных.

В третьей главе описан процесс реализации и тестирования ПО информационной системы для анализа медицинских данных.

В заключении описываются результаты выполнения выпускной квалификационной работы.

Бакалаврская работа состоит из 45 страниц текста, 24 рисунков, 9 таблиц и 24 источников.

Глава 1 Постановка задачи на разработку программного обеспечения информационной системы для анализа медицинских данных

1.1 Программное обеспечение медицинской информационно-аналитической системы

Медицинские информационно-аналитические системы (МИАС) предназначены для сбора, компилирования и анализа данных о состоянии здоровья людей, что помогает управлять здоровьем населения и сокращать расходы на здравоохранение.

Основная задача МИАС – облегчить использование и обмен данными, связанными со здоровьем, чтобы лица, принимающие решения, могли выявлять проблемы, делать обоснованный выбор в отношении политики здравоохранения и максимизировать распределение ресурсов [5].

МИАС разрабатываются и внедряются для улучшения результатов лечения пациентов и обеспечения эффективного использования ограниченных ресурсов.

Рассмотрим функциональные особенности ПО МИАС.

Достижения в области технологий создания и сбора данных привели к огромному росту данных в базах данных здравоохранения. ПО для управления пациентами, медицинское оборудование, программное обеспечение для клинического анализа и медицинской визуализации – это лишь несколько примеров этих технологий.

При ежедневном функционировании всех этих различных программных продуктов для здравоохранения исследователи сталкиваются с неуправляемым масштабом данных (объемом).

Поэтому наличие опции интеллектуального анализа данных (Data mining), способного обрабатывать большие объемы данных, имеет решающее значение для анализа медицинских данных.

«Существует несколько типов ПО МИАС:

- инструменты интеллектуального анализа данных, которые могут самостоятельно выполнять все этапы методологии интеллектуального анализа медицинских данных (инструменты сквозного анализа);
- инструменты, представляющие собой библиотеки, выполняющие задачи интеллектуального анализа данных с помощью технологий ты машинного обучения;
- инструменты для статистики и вычислений;
- инструменты для предварительной обработки и анализа крупномасштабных данных» [16].

ПО МИАС должно обеспечивать возможности исследования и визуализации медицинских данных, которые могут способствовать процессу лучшего понимания полученных результатов.

1.2 Формирование требований на разработку программного обеспечения информационной системы для анализа медицинских данных

Как было отмечено выше, по своим функциональным и архитектурным особенностям ПО информационной системы для анализа медицинских данных относится к категории ПО МИАС.

Для разработки требований к ПО информационной системы для анализа медицинских данных используем методологию FURPS+.

«Преимущества использования FURPS+ заключаются в следующем:

- способствует более полному и точному определению требований к системе, а также их приоритезации, согласованию и верификации» [6];
- облегчает коммуникацию и сотрудничество между заинтересованными сторонами, такими как заказчики, разработчики,

тестировщики, пользователи и др.;

- поддерживает использование различных методологий и инструментов для управления требованиями.

На основании проведенного анализа выработаны требования к ПО информационной системы для анализа медицинских данных по методологии FURPS+, представленные в таблице 1.

Таблица 1 – Требования к ПО информационной системы для анализа медицинских данных

«Требование	Статус	Полезность	Риск	Стабильность
Functionality – Функциональные требования				
Анализ медицинских данных	одобренное	средняя	средний	низкая
Формирование отчетов с визуализацией результатов	одобренное	критическая	средний	низкая
Usability– Требования к удобству использования				
Современный дизайн	одобренное	критическая	средний	низкая
Простота проведения анализа	одобренное	критическая	средний	низкая
Reliability– Требования к надежности				
Допустимая частота/периодичность сбоев: 1 раз в 300 часов	одобренное	важная	средний	средняя
Среднее время сбоев: 1 раб. день	одобренное	важная	средний	средняя
Возможность восстановления системы после сбоев: 1 раб. день	одобренное	важная	средний	средняя
Режим работы: рабочий день	одобренное	важная	средний	средняя
Performance – Требования к производительности				
Допустимое количество одновременно работающих пользователей: 1	предложенное	важная	средний	средняя
Время реакции на возникновение аварийной ситуации: 1 с	предложенное	важная	средний	средняя
Время устранения критических проблем: в течение рабочего дня	предложенное	важная	средний	средняя» [20]

Продолжение таблицы 1

«Требование	Статус	Полезность	Риск	Стабильность
Проектные ограничения				
Использование методов МО	предложенное	важная	средний	средняя
Отсутствие избыточного функционала	предложенное	важная	средний	средняя
Низкая совокупная стоимость владения	предложенное	критическая	средний	низкая» [20]

Разработанный перечень требований является основой для разработки ПО информационной системы для анализа медицинских данных.

1.3 Обзор и анализ аналогов программного обеспечения информационной системы для анализа медицинских данных

1.3.1 Платформа для анализа данных RapidMiner

«RapidMiner – это бесплатный программный инструмент с открытым исходным кодом для интеллектуального анализа данных и текста.

Используется для анализа данных в медицине.

Помимо операционных систем Windows, RapidMiner также поддерживает системы Macintosh, Linux и Unix. Он доступен как отдельное приложение для анализа данных/текста, а также как механизм интеллектуального анализа данных/текста для интеграции в собственные продукты пользователя» [9].

Основными преимуществами RapidMiner:

- инструменты и функции RapidMiner предлагают пользователям мощные возможности и в то же время представлены в удобном интерфейсе, который позволяет пользователям с самого начала продуктивно выполнять свою работу;
- платформа предоставляет пользователям правильный набор

инструментов, которые позволяют эффективно использовать даже самые неорганизованные, лаконичные и, казалось бы, бесполезные данные.

На рисунке 1 показан пример отчета с визуализацией результатов анализа RapidMiner.

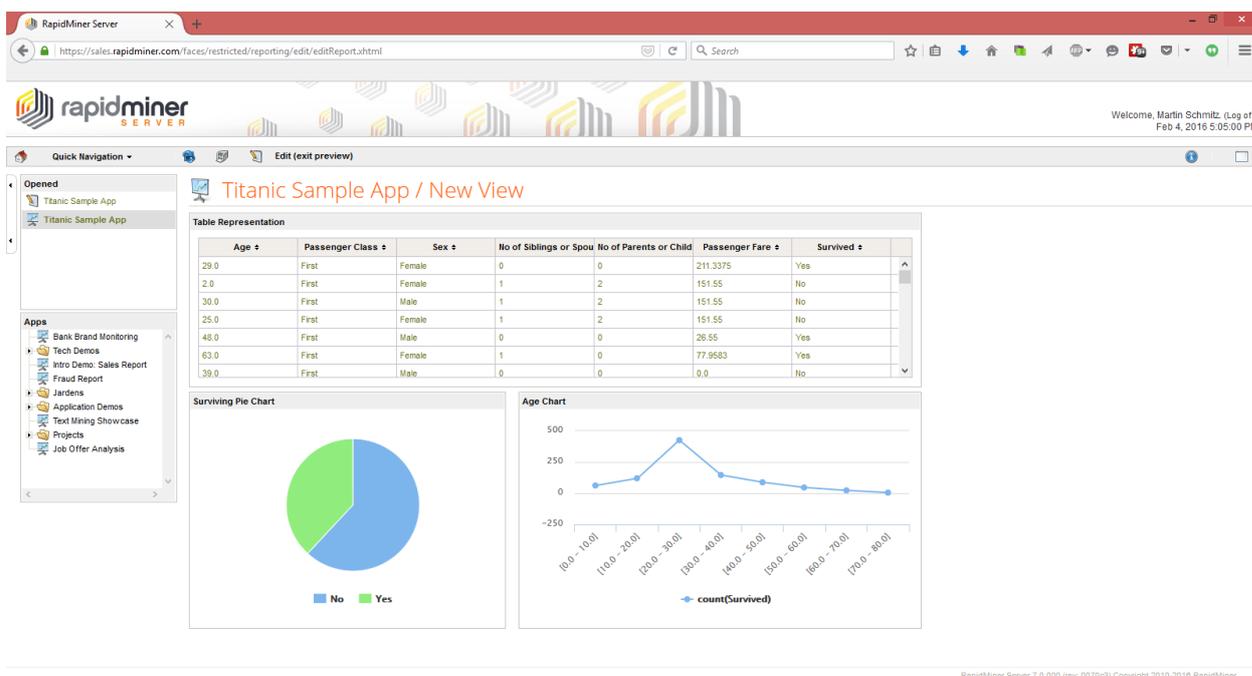


Рисунок 1 – Отчет с визуализацией результатов анализа RapidMiner

RapidMiner имеет некоторые ограничения. Перед применением моделей требуется предварительная обработка и разделение данных.

Кроме того, платформа может иметь ограничения с точки зрения прогнозирующей производительности, например, в оценке методов машинного обучения для анализа настроений.

1.3.2 Аналитическая платформа Logiom

Logiom – это российская аналитическая low-code платформа, которая позволяет проводить анализ данных любого уровня сложности без программирования [1]. Используется для анализа данных в медицине.

На рисунке 2 показан пример отчета с визуализацией результатов анализа Logitom.

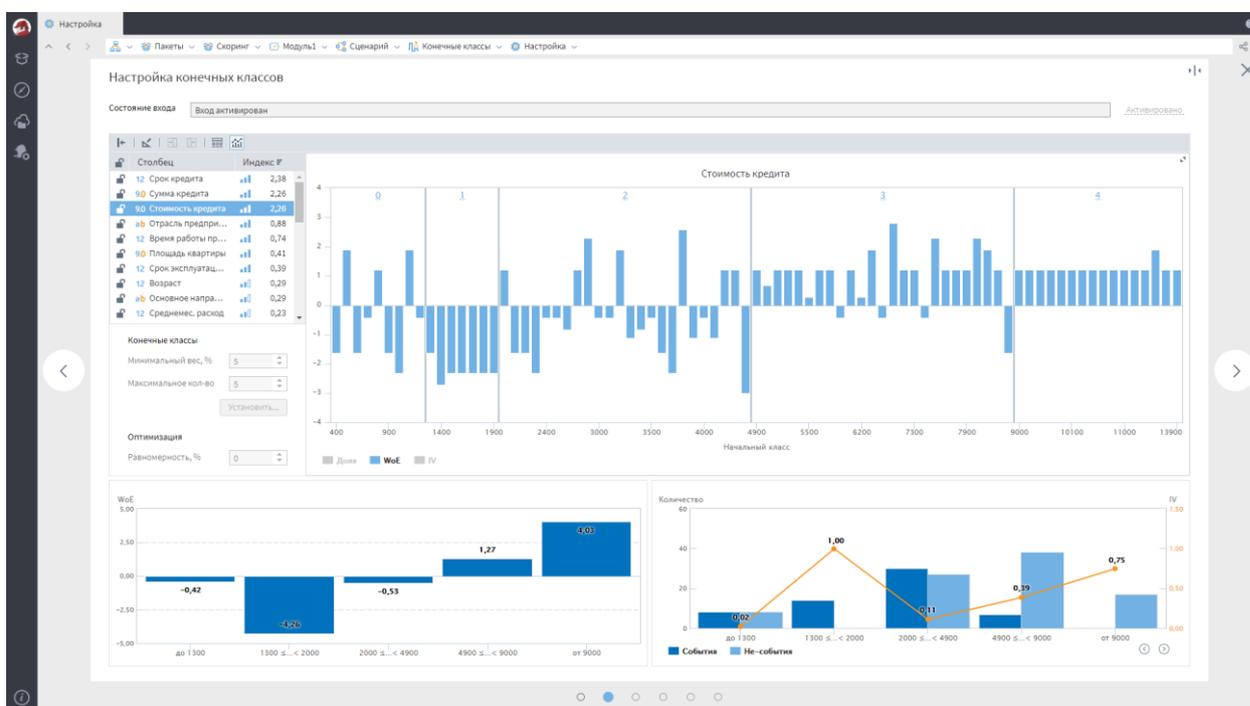


Рисунок 2 – Отчет с визуализацией результатов анализа Logitom

Функциональные возможности:

- визуальное проектирование;
- дружелюбный веб-интерфейс;
- подключение пакетов с готовыми бизнес-компонентами;
- высокая производительность;
- использование методов МО и др.

К недостаткам программы относят необходимость кодирования для нетривиальной логики и ограниченный функционал анализа.

Имеется бесплатная версия.

1.3.3 Аналитическая платформа KNIME

Knime Analytics Platform – open source фреймворк для анализа данных.

Данный фреймворк позволяет реализовывать полный цикл анализа

данных включающий чтение данных из различных источников, преобразование и фильтрацию, собственно анализ, визуализацию и экспорт [13].

Функциональные возможности:

- доступ, смешивание и преобразование данных из любого источника;
- визуализация и анализ;
- возможность сохранить, поделится и использовать повторно;
- повышение квалификации и др.

Для анализа данных доступны различные статистические методы (включая линейную корреляцию, проверку гипотез) а также Data Mining методы (например линейная регрессия, построение decision trees, cluster view).

На рисунке 3 показан график результатов анализа в Knime.

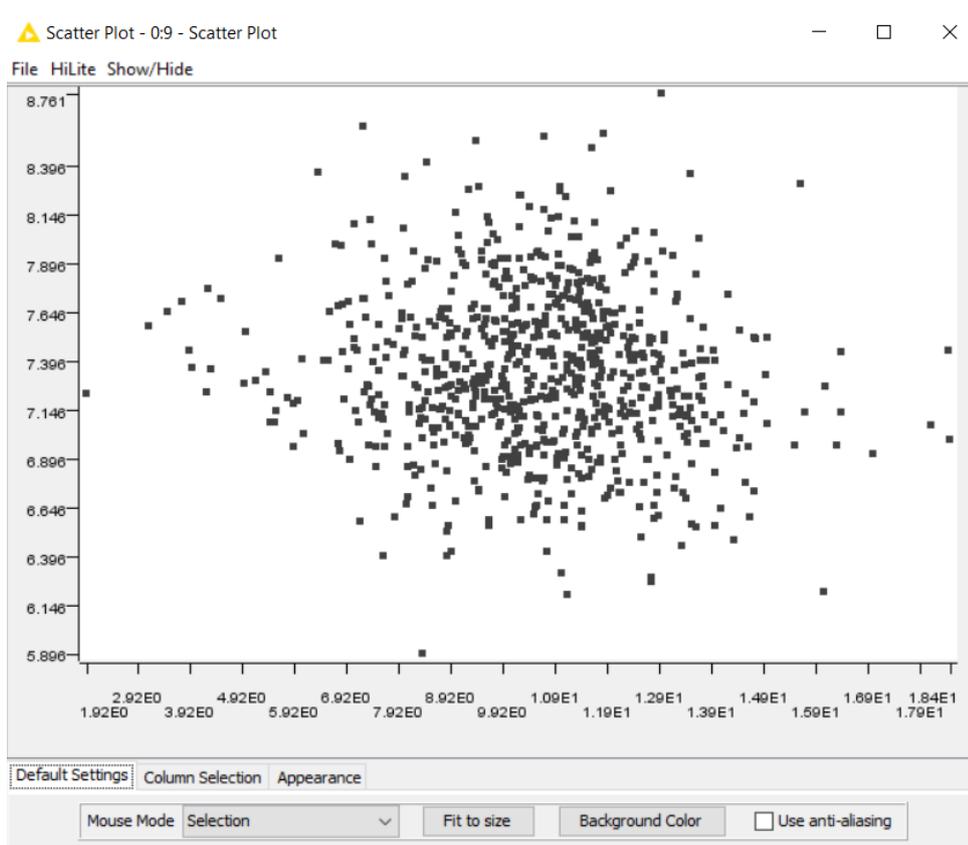


Рисунок 3 – График результатов анализа Knime

К недостаткам программы относят недостаточно гибкое поведение узлов и сложность программирование некоторых задач.

Для анализа характеристик рассмотренных аналогов

Таблица 2 – Сравнение характеристик аналогов ПО МИАС

Характеристика/балл (0-3)	RapidMiner	Loginom	Knime
Анализ медицинских данных	3	3	3
Формирование отчетов с визуализацией результатов	3	3	3
Простота проведения анализа	1	1	1
Использование методов МО	3	3	3
Отсутствие избыточного функционала	1	1	1
Низкая совокупная стоимость владения	3	3	3
Итого	14	14	14

Таким образом, сформулированным требованиям к ПО МИАС не соответствует ни одна из представленных программ.

Главный недостаток рассмотренного ПО заключается в сложности проведения анализа и избыточной функциональности.

Поэтому представляет интерес разработка специального ПО МИАС, предназначенного для анализа медицинских данных.

Выводы к главе 1

Результаты выполнения первой главы позволили сделать следующие выводы:

- по своим функциональным и архитектурным особенностям ПО информационной системы для анализа медицинских данных относится к категории ПО МИАС;

- ПО МИАС должно обеспечивать возможности исследования и визуализации медицинских данных, которые могут способствовать процессу лучшего понимания полученных результатов;
- ПО МИАС должно обеспечивать интеллектуальный анализ данных (Data mining), способный обрабатывать большие объемы данных с помощью технологий МО;
- сформулированным требованиям к ПО МИАС не соответствует ни одна из представленных программ;
- главный недостаток рассмотренного ПО заключается в сложности проведения анализа и избыточной функциональности;

Поэтому представляет интерес разработка специального ПО МИАС, предназначенного для анализа медицинских данных.

Глава 2 Проектирование программного обеспечения информационной системы для анализа медицинских данных

2.1 Выбор методологии проектирования программного обеспечения информационной системы для анализа медицинских данных

Методологии разработки программного обеспечения являются основой для разработки и поддержки ПО. Они варьируются от строго предписывающих методов, таких как Agile и Scrum, до весьма общих, таких как методология водопада. Спектр вариантов может затруднить выбор процесса [10].

Однако выбор методологии требует рассмотрения нескольких факторов, включая требования к ПО, опыт и навыки разработки.

В основу каждой методологии разработки ПО положено понятия жизненного цикла ПО, который состоит из нескольких этапов.

Первый этап жизненного цикла разработки ПО, как правило, включает в себя планирование и требует наличия менеджера проекта, который отслеживает все задачи и может предсказать, сколько времени потребуется для выполнения каждой из них. Начальный этап планирования также включает в себя анализ требований, при котором менеджеры проектов сотрудничают с персоналом клиента, чтобы документировать, что необходимо сделать. Кроме того, они смотрят на затраты на персонал и другие ресурсы.

На втором этапе разработчики преобразуют требования в практические задачи. Тестирование начинается на третьем и четвертом этапах после завершения написания компьютерных программ, основанных на этих целях.

Разработчики обычно передают завершённые программные проекты группам обеспечения качества на пятом этапе. Эти специалисты гарантируют, что программа работает должным образом, прежде чем передать ее клиентам

или сотрудникам компании, которые используют ее в своей повседневной работе.

Рассмотрим и сравним основные методологии разработки ПО: Waterfall, Scrum и RUP.

Водопадный подход (Waterfall) или каскадная модель разработки ПО – это последовательный, простой и систематический процесс, представленный на рисунке 4.

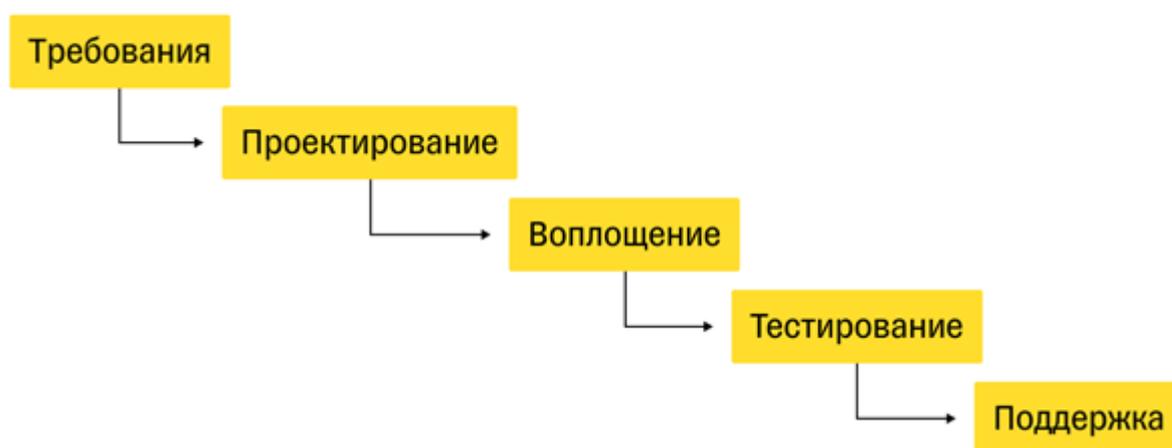


Рисунок 4 – Схема процесса разработки ПО в методологии водопада

При разработке ПО типичный водопадный процесс начинается с этапа планирования, на котором определяются и анализируются требования проекта для определения оптимального решения [24].

Следующий шаг – проектирование, за которым следуют этапы разработки и тестирования.

Наконец, есть реализация и поддержка производства. Этот метод работы придает структуру более крупным проектам, но может занять довольно много времени, поскольку включает в себя множество этапов.

Поскольку команда разработчиков должна завершить каждый шаг, прежде чем перейти к следующему, любые изменения на этом пути требуют значительной переработки предыдущих шагов, что в конечном итоге

приводит к задержке поставки и увеличению затрат для всех участвующих сторон.

Scrum – это методология разработки ПО, которая помогает командам более эффективно работать вместе и достигать целей проекта [18].

Жизненный цикл (ЖЦ) проекта разработки ПО в методологии Scrum показан на рисунке 5.



Рисунок 5 – ЖЦ проекта разработки ПО в методологии Scrum

Существует множество различных вариантов Scrum.

Однако у них есть общие характеристики: роли, встречи, артефакты и правила.

Роли включают владельца продукта, Scrum-мастера Scrum и членов команды разработчиков и тестировщиков.

У артефактов есть бэклог, бэклог спринта, эпики, списки функций и задачи. Правила охватывают распределение ответственности за задачи, а также методы управления временем, такие как диаграммы сгорания.

Основная цель методологии RUP (Rational Unified Process) – предоставить модель для эффективной реализации коммерчески проверенных

подходов к разработке для использования на протяжении всего жизненного цикла разработки ПО [2].

Следует отметить, что RUP – это не конкретная модель разработки, а скорее адаптивная и адаптированная к конкретным потребностям вашего проекта, команды или организации. RUP основана на нескольких фундаментальных идеях, таких как этапы разработки и строительные блоки, которые определяют, кто, что, когда и как будет происходить разработка.

RUP снижает вероятность непредвиденных обстоятельств. затраты на разработку и предотвращение нерационального использования ресурсов.

RUP предлагает итеративный подход к проектированию и разработке ПО, основанный на спиральном жизненном цикле.

Жизненный цикл ПО в методологии RUP представлен на рисунке 6.

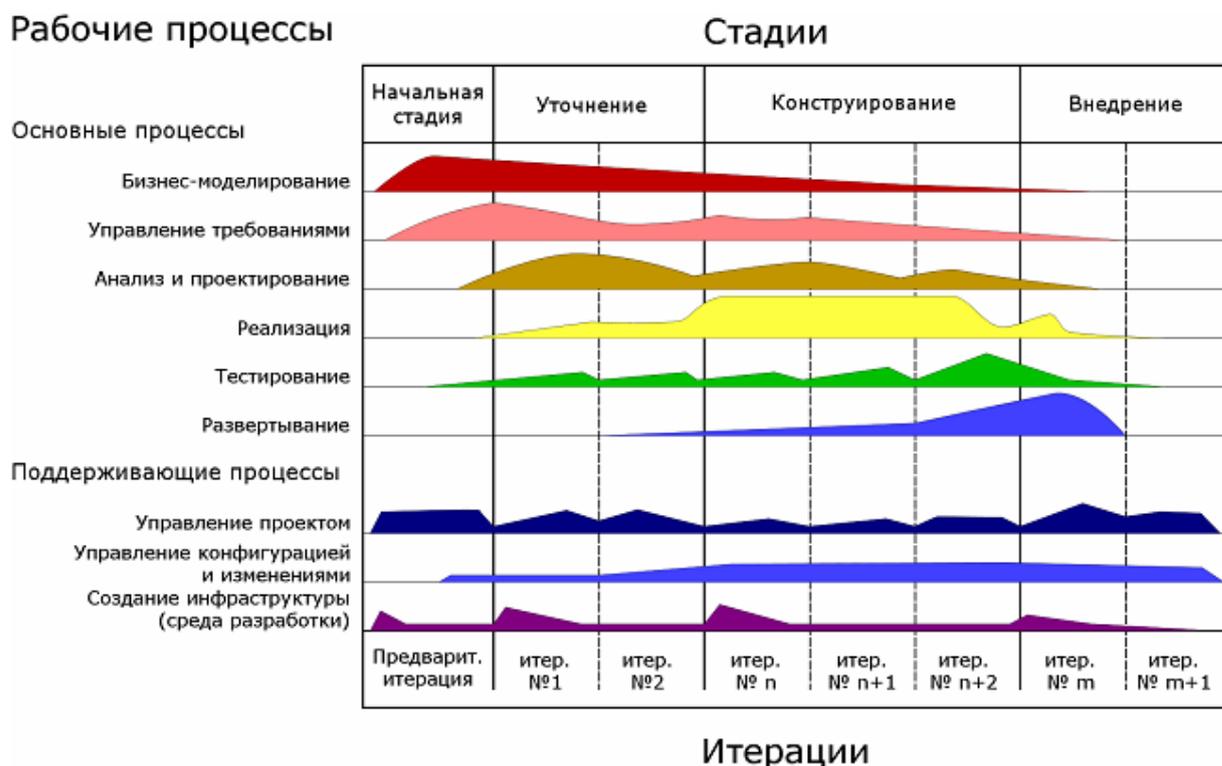


Рисунок 6 – Жизненный цикл ПО в методологии RUP

Для сравнения рассмотренных методологий используем таблицу 3.

Таблица 3 – Сравнение методологий разработки ПО

Методология разработки ПО	Достоинства	Недостатки
Waterfall	Отличная документация; плавная интеграция новых сотрудников. Высокоточная оценка затрат на разработку. Поддерживает этап тестирования. Конечный продукт соответствует ожиданиям.	Менее гибкий процесс. Более длительный срок доставки.
Scrum	Легкое устранение ошибок; подходит для динамичных проектов. Удобная процедура тестирования. Улучшает мотивацию команды Гибкость.	Строгое управление. Не подходит для большой команды.
RUP	Использует лучшие части модели Waterfall и включает их в более итеративный процесс, позволяющий вносить изменения. Подходит для небольших проектов.	Как и Waterfall, RUP также требует много процессов и может слишком сильно полагаться на отзывы заинтересованных сторон. Даже будучи итеративным процессом, он может быть слишком медленным для некоторых типов проектов.

По результатам сравнения достоинств и недостатком рассмотренных методологий выбираем для разработки ПО методологию RUP.

Главным преимуществом методологии RUP для рассматриваемого случая является возможность применения для небольших проектов.

2.2 Логическое моделирование программного обеспечения информационной системы для анализа медицинских данных

Одним из основных этапов начальной фазы проектирования RUP является логическое моделирование ПО.

Логическое моделирование – это важнейший процесс, который может принести пользу, как бизнесу, так и заинтересованным сторонам в сфере ИТ.

Логические модели обеспечивают общий язык для этих двух групп для эффективного общения, гарантируя, что все участники проекта разработки ПО понимают цели и требования. Такое общее понимание помогает устранить недоразумения и гарантирует, что все стороны работают над достижением одних и тех же целей.

Логические модели способствуют эффективной реализации ПО, позволяя разработчикам детализировать сложные проекты разработки ПО на более мелкие части в соответствии со спецификациями, что упрощает работу команд, работающих на разных уровнях, поскольку им не нужны обширные технические знания благодаря предоставленным четким описаниям в логических схемах.

Кроме того, после реализации логическое моделирование дает преимущества, выходящие за рамки задач разработки.

Пользователи ПО могут непрерывно отслеживать работу процессов на основе логической схемы, созданной ранее на этапах планирования, что позволяет своевременно выявлять ошибки до увеличения затрат, связанных с исправлением после поставки.

Логическая модель ПО позволяет быстро принимать решения при возникновении изменений, поскольку показывает, где какие-либо конкретные изменения могут повлиять на другие компоненты, что сводит к минимуму негативные последствия, такие как простои или потеря данных, в целом повышая общую эффективность во всех областях деятельности подразделений. Это в значительной степени связано со структурированием рабочих процессов вокруг хорошо организованных объектов, представленных графически посредством визуального представления, обеспечиваемого диаграммами UML. Логическое моделирование может оказать существенное влияние на бюджет и сроки проекта. Предоставляя четкое понимание того, что

необходимо сделать, это помогает исключить ненужную работу или действия, которые могут привести к дополнительным затратам.

Логическая модель отличается от концептуальной модели, которая отображает предметную область в форме взаимосвязанных объектов без указаний способов их физического хранения, и от физической модели, которая является материальным воплощением логической модели в виде программного обеспечения, баз данных, устройств и т.п.

2.2.1 Разработка функциональной модели ПО.

На начальной фазе проектирования RUP определяются основные требования, ограничения и ключевая функциональность продукта. Создается базовая версия модели прецедентов.

Цель – определение основной функциональности ПО автоматизации тестирования бизнес-приложений.

Ключевые участники – бизнес-аналитик и программист.

Входная информация: перечень функциональных требований к ПО, методы анализа медицинских данных.

Результат – функциональная модель ПО МИАС, представленная в виде диаграммы вариантов использования [7].

В процессе разработки диаграммы вариантов использования ПО был выделены актор Аналитик медицинских данных.

Варианты использования (прецеденты) ПО представлены в таблицах 4 - 8.

Таблица 4 – Описание прецедента: Выбор датасета

Элемент диаграммы	Описание
Прецедент	Выбор датасета
ID	1
Краткое описание	Выбор входного набора данных для анализа
Главный актор	Аналитик медицинских данных
Второстепенный актор	Нет» [21]

Продолжение таблицы 4

Элемент диаграммы	Описание
Предусловие	Нет
Основной поток	Аналитик медицинских данных выбирает входной датасет
Постусловие	Нет
Альтернативные потоки	Нет» [21]

Таблица 5 – Описание прецедента: Подготовка датасета

«Элемент диаграммы	Описание
Прецедент	Подготовка датасета
ID	2
Краткое описание	Подготовка датсета для МО
Главный актер	Аналитик медицинских данных
Второстепенный актер	Нет
Предусловие	Нет
Основной поток	Аналитик медицинских данных готовит датасет для МО
Постусловие	Нет
Альтернативные потоки	Нет» [21]

Таблица 6 – Описание прецедента: Проведения анализа данных

«Элемент диаграммы	Описание
Прецедент	Проведение анализа данных
ID	3
Краткое описание	Проведение анализа медицинских данных
Главный актер	Аналитик медицинских данных
Второстепенный актер	Нет
Предусловие	Нет
Основной поток	Аналитик медицинских данных проводит анализ данных
Постусловие	Нет
Альтернативные потоки	Нет» [21]

Таблица 7 – Описание прецедента: Визуализация результатов анализа

«Элемент диаграммы	Описание
Прецедент	Визуализация результатов анализа
ID	4
Краткое описание	Создание графиков и гистограмм
Главный актер	Аналитик медицинских данных
Второстепенный актер	Нет
Предусловие	Нет
Основной поток	Аналитик медицинских данных создает графики и гистограммы по данным результатов анализа
Постусловие	Нет
Альтернативные потоки	Нет» [21]

Таблица 8 – Описание прецедента: Формирование аналитического отчета

«Элемент диаграммы	Описание
Прецедент	Формирование аналитического отчета
ID	5
Краткое описание	Формирование отчета результатов анализа
Главный актер	Аналитик медицинских данных
Второстепенный актер	Нет
Предусловие	Нет
Основной поток	Аналитик медицинских данных формирует аналитический отчет
Постусловие	Нет
Альтернативные потоки	Нет» [21]

Для разработки диаграмм UML использован онлайн-сервис Visual Paradigm [23]

Разработанная диаграмма вариантов использования ПО информационной системы для анализа медицинских данных показана на рисунке 7.



Рисунок 7 – Диаграмма вариантов использования ПО информационной системы для анализа медицинских данных

Показанная диаграмма представляет функциональную модель разрабатываемого ПО.

2.2.2 Разработка архитектуры ПО

На фазе уточнения RUP создается архитектура ПО.

Цель – разработка программной архитектуры системы для обеспечения стабильной основы для части работ по проектированию и внедрению.

Ключевые участники – аналитик медицинских данных и разработчик.

Входная информация: модель вариантов использования ПО.

Результат – архитектура ПО.

Архитектура ПО относится к набору структур, необходимых для построения программной системы. Каждая структура включает элементы программного обеспечения, их отношения и свойства, связанные как с элементами, так и с отношениями.

Для представления архитектуры ПО используем диаграмму компонентов UML [22].

Диаграмма компонентов – это вид диаграммы UML, которая компоненты системы и их взаимодействие. В ИСУ компонентами системы являются подсистемы, из которых она состоит.

Диаграмма компонентов ПО МИАС представлена на рисунке 8.



Рисунок 8 – Диаграмма компонентов ПО информационной системы для анализа медицинских данных

Архитектура ПО служит основой для всего проекта разработки, позволяя руководству проекта экстраполировать необходимые задачи для участвующих команд.

Выводы по главе 2

Результаты проделанной работы позволили сделать следующие выводы:

- главным преимуществом методологии RUP для рассматриваемого случая является возможность применения для небольших проектов;
- логическое моделирование ПО в значительной степени связано со структурированием рабочих процессов вокруг хорошо организованных объектов, представленных графически посредством визуального представления, обеспечиваемого диаграммами UML;
- на начальной фазе проектирования RUP определяются основные требования, ограничения и ключевая функциональность продукта. Создается базовая версия модели прецедентов.
- на фазе уточнения RUP создается архитектура ПО.

Архитектура ПО служит основой для всего проекта разработки, позволяя руководству проекта экстраполировать необходимые задачи для участвующих команд.

Глава 3 Реализация и тестирование программного обеспечения информационной системы для анализа медицинских данных

3.1 Выбор средства разработки программного обеспечения информационной системы для анализа медицинских данных

Для реализации ПО выбран язык Python.

Для выбора среды разработки сравним характеристики двух сред разработки: Replit и Python IDLE.

Рассмотрим возможности среды Replit [15].

Replit – бесплатная интегрированная среда разработки для совместной работы в браузере для написания кода на Python, C++, HTML, CSS и другие.

Пользователи, которые обучают программированию или переключаются между несколькими компьютерами, находят Replit очень полезным, поскольку он позволяет им работать над своими проектами без необходимости устанавливать среду разработки на каждый компьютер, который они используют. Удобный интерфейс IDE в сочетании с ее надежной функциональностью делает программирование более доступным и эффективным.

Преимущества и недостатки среды Replit:

- облачная доступность;
- поддержка нескольких языков программирования;
- простота использования;
- возможности совместной работы;
- кодирование с помощью искусственного интеллекта;
- проблемы с производительностью;
- ограниченные инструменты отладки;
- не хватает некоторых расширенных функций IDE;
- ограниченные возможности развертывания.

Рассмотрим возможности среды Python IDLE [14].

Python IDLE – интегрированная среда разработки с языком программирования Python и базовым интерфейсом, который позволяет пользователям писать, выполнять и отлаживать код Python.

Python IDLE имеет несколько функций, которые делают его популярным среди начинающих и опытных разработчиков Python:

- базовый пользовательский интерфейс;
- инструменты отладки;
- подсветка синтаксиса;
- автозаполнение и др.

Преимущества и недостатки среды Python IDLE:

- простота и скорость;
- интегрированные инструменты отладки;
- ограниченные возможности и функциональность;
- нет поддержки совместной работы;
- ограниченная поддержка визуализации данных;
- нет поддержки языка разметки;
- нет интеграции с другими языками программирования.

Для сравнения рассмотренных сред программирования используем таблицу 9.

Таблица 9 – Сравнение характеристик сред разработки на языке Python

Характеристика	Replit	Python IDLE
Интерфейс	Веб-интерфейс	Десктоп-приложение
Вывод	В консоль, окна Output и Webview	В отдельном окне консоли
Совместная работа	Встроенная поддержка совместного использования и совместной работы	Нет встроенной функции совместной работы
Организация кода	Код хранится в папке проекта	Код записан в одном файле

Продолжение таблицы 9

Характеристика	Replit	Python IDLE
Отладка	Доступна поддержка отладки	Доступна поддержка отладки
Визуализация	Хорошая поддержка	Ограниченная поддержка
Кривая обучения	Легко изучается	Легко изучается
Варианты использования	Для анализа данных, веб-разработки и прототипирования	Написание небольших скриптов и изучение синтаксиса Python

По результатам сравнения и с учетом предпочтений разработчика выбираем среду разработки Replit.

Для поддержки машинного обучения используем библиотеку Scikit-learn.

«Scikit-learn – это библиотека анализа данных с открытым исходным кодом и золотой стандарт машинного обучения в экосистеме Python.

Ключевые концепции и функции включают в себя алгоритмические методы принятия решений, в том числе:

- классификация: идентификация и категоризация данных на основе закономерностей;
- регрессия: прогнозирование или прогнозирование значений данных на основе среднего значения существующих и запланированных данных;
- кластеризация: автоматическая группировка похожих данных в наборы данных;
- взаимодействие с библиотеками numPy, pandas и matplotlib и др.

Scikit-learn включает алгоритмы, поддерживающие прогнозный анализ, которые варьируются от простой линейной регрессии до алгоритмов деревьев решений» [17].

3.2 Реализация программного обеспечения информационной системы для анализа медицинских данных

Разработана диаграмма классов ПО.

Диаграмма классов относится к статическому виду диаграмм UML, который описывает структуру системы, а не ее поведение. Диаграмма классов помогает анализировать, проектировать и документировать программные системы. На диаграмме классов классы представляются в виде прямоугольников, разделенных на три части: имя класса, атрибуты и операции. Отношения между классами показываются с помощью линий, которые могут иметь разные символы на концах, обозначающие тип и кратность отношения. Например, ассоциация, агрегация, композиция, наследование, реализация и т.д. [11].

Диаграмма классов ПО показана на рисунке 9.

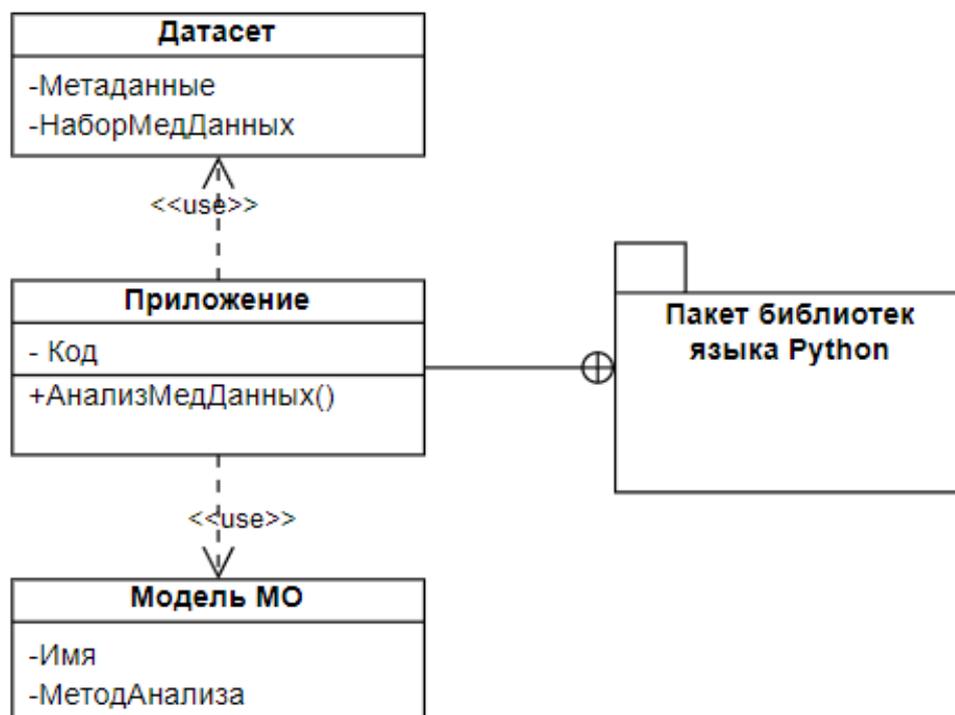


Рисунок 9 – Диаграмма классов ПО информационной системы для анализа медицинских данных

В среде Replit разработан код ПО.

На рисунке 10 показан код импорта библиотек Python и загрузки входного датасета [3].

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import pathlib
import tkinter
from tkinter import filedialog as fd
from pathlib import Path
#Выбор датасета
selected_file = Path(fd.askopenfile(mode='r', filetypes=[('Datasets',
 '*.csv')]).name)
print(type(selected_file), selected_file)
p=selected_file
df=pd.read_csv(p)
print(df.head())
p = pathlib.Path('Sleep_health_and_lifestyle_dataset2_2.csv')
ds = pd.read_csv(p)
```

Рисунок 10 – Код импорта библиотек Python и загрузки входного датасета

Далее выполняется подготовка данных (очистка, трансформации и предподготовка).

Результатом является создание датафрейма для анализа данных.

На рисунке 11 показан код подготовки данных.

```
#замена значений None на значение 1
ds = ds.fillna(1)
#Замена значений Sleep Apnea на значение 2
ds.loc[ds["SD"] == "Sleep Apnea", 'SD'] = 2
#Замена значений Insomnia на значение 3
ds.loc[ds["SD"] == "Insomnia", 'SD'] = 3
ds["SD"] = ds["SD"].astype(int)
#Убрать дубликаты
ds.drop_duplicates()
ds
#Переименование столбцов
ds = ds.rename(columns={"Quality of Sleep": "QS",
                        "Physical Activity Level": "PAL",
                        "Stress Level": "SL", "BMI Category": "BMIC",
                        "Sleep Disorder": "SD"})
```

Рисунок 11 – Код подготовки данных

Разработан код для анализа данных и визуализации результатов.
На рисунке 12 показан код диаграммы «Ящик с усами».

```
#Ящички с усами  
plt.boxplot(ds)  
plt.show()
```

Рисунок 12 – Код диаграммы «Ящик с усами»

В описательной статистике диаграмма «Ящик с усами» (boxplot diagram) представляет собой тип диаграммы, часто используемый в поясняющем анализе данных. Такие диаграммы визуально показывают распределение числовых данных и асимметрию путем отображения квартилей (или процентилей) и средних значений данных. Диаграмма «Ящик с усами» широко применяется в доказательной медицине [4].

На рисунке 13 показан код тепловой карты.

```
#Тепловая карта  
plt.figure(figsize=(10,5))  
c= ds.corr()  
sns.heatmap(c, cmap="YlGnBu", annot=True)  
plt.title('Тепловая карта признаков набора данных')  
plt.show()
```

Рисунок 13 – Код тепловой карты

Тепловая карта (Heat Map) – это графическое представление данных, в котором значения в точках набора данных представлены цветами. Это помогает визуально выделить области с высокой или низкой концентрацией, что делает их анализ более интуитивным и понятным для принятия

управленческих решений.

На рисунке 14 показан код гистограммы распределения по группам заболеваний.

```
#Гистограммы распределения по группам
ds.SD.value_counts().nlargest(100).plot(kind='bar', figsize=(10,5),color='red')
plt.title("Распределение по группам с нарушением сна")
plt.ylabel('Количество респондентов',color='blue')
plt.xlabel('Группа с нарушением сна',color='green')
plt.show()
ds.QS.value_counts().nlargest(100).plot(kind='bar') #GH
plt.title("Распределение респондентов по качеству сна")
plt.ylabel('Количество респондентов')
plt.xlabel('Качество сна');
plt.show()
```

Рисунок 14 – Код гистограммы распределения по группам заболеваний

На рисунке 15 показан код построения точечной диаграммы.

```
# выведем точечную диаграмму
plt.scatter(ds['BMI'], ds['SD'], color = 'b', label = "Нарушение сна")
plt.xlabel("Индекс массы тела")
plt.ylabel("Нарушение сна")
plt.show()
```

Рисунок 15 – Код построения точечной диаграммы

Разработан код анализа данных с помощью моделей МО для Data mining [8].

Для создания моделей использована библиотека Scikit-Learn.

На рисунке 16 показан код построения классификатора данных.

```

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(
    ds.iloc[:, :-1],
    ds.iloc[:, -1],
    test_size = 0.20
)
X_train.shape, X_test.shape, y_train.shape, y_test.shape
X_train.head()
y_train.head()
#выполним классификацию
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier()
classifier.fit(X_train, y_train)
#построим дерево решений
from sklearn import tree
tree.plot_tree(classifier)
plt.show()

```

Рисунок 16 – Код построения дерева решений

Представленный код был успешно апробирован в среде Replit.

3.3 Тестирование программного обеспечения информационной системы для анализа медицинских данных

Для тестирования разработанного ПО использован метод функционального тестирования [12].

Функциональное тестирование гарантирует, что программное обеспечение соответствует своим требованиям. Это делается путем тестирования каждой отдельной функции программного обеспечения, чтобы убедиться, что оно работает должным образом.

На рисунке 17 показано главное окно приложения.

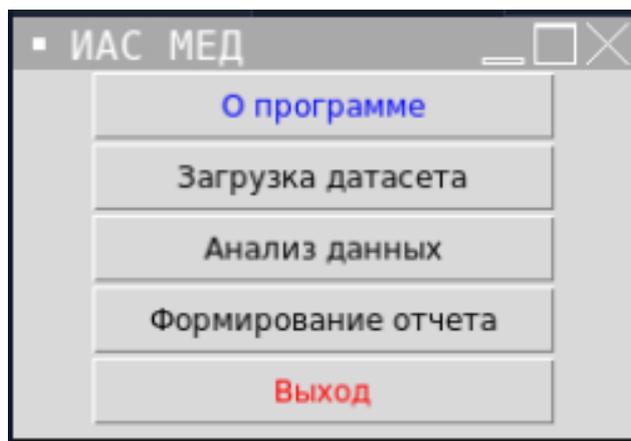


Рисунок 17 – Главное окно приложения

При нажатии на кнопку «Вход» открывается диалоговое окно выбора файла датасета (рисунок 18).

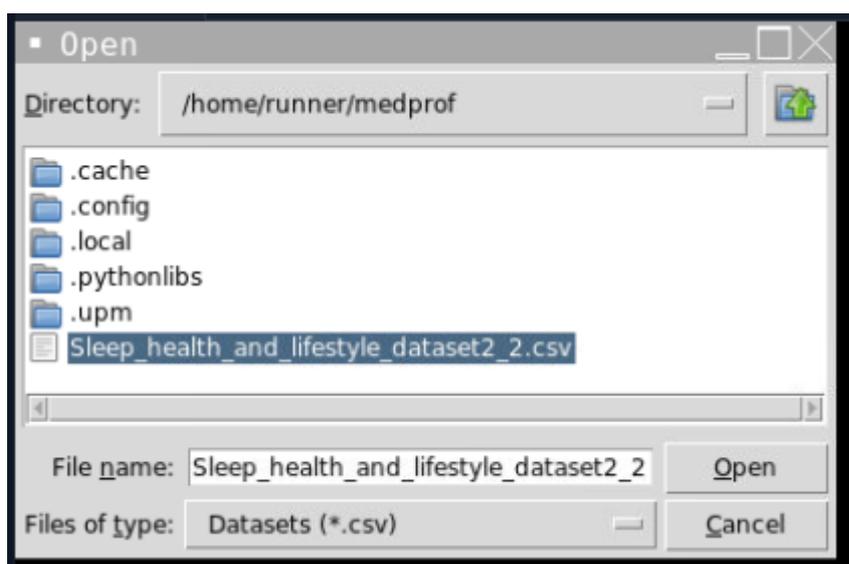


Рисунок 18 – Окно выбора файла датасета

В качестве входного датасета использован свободно распространяемый датасет «Sleep Health and Lifestyle».

«Датасет состоит из 400 строк и 13 столбцов, охватывающих широкий спектр переменных, связанных со сном и повседневными привычками.

Он включает в себя такие детали, как пол, возраст, род занятий, продолжительность сна, качество сна, уровень физической активности, уровень стресса, категория индекса массы тела (ИМТ), артериальное давление, частота сердечных сокращений, ежедневные шаги, а также наличие или отсутствие нарушений сна» [19].

На рисунке 19 показана диаграмма «Ящик с усами» для признака «Уровень физической активности» (PAL, Physical Activity Level).

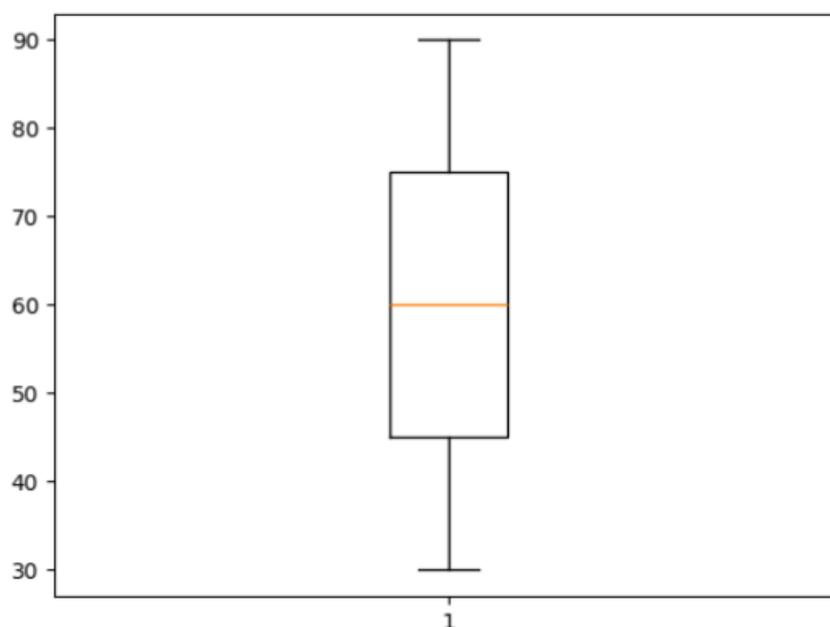


Рисунок 19 – Диаграмма «Ящик с усами» для признака «Уровень физической активности»

На рисунке 20 показана тепловая карта признаков набора данных.



Рисунок 20 – Тепловая карта признаков набора данных

На рисунке 21 показана гистограмма распределения пациентов по уровню стресса.

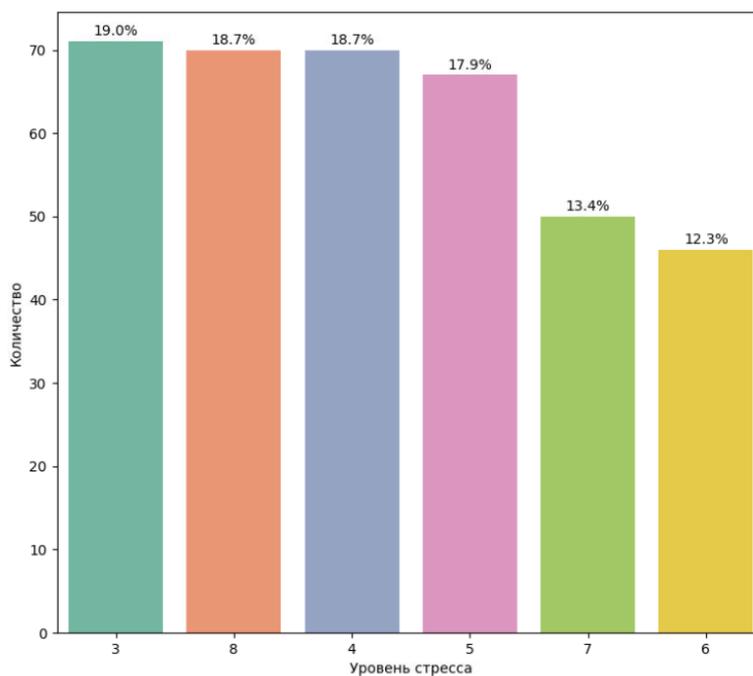


Рисунок 21 – Распределение пациентов по уровню стресса

На рисунке 22 показана точечная диаграмма зависимости нарушения сна от ИМТ.

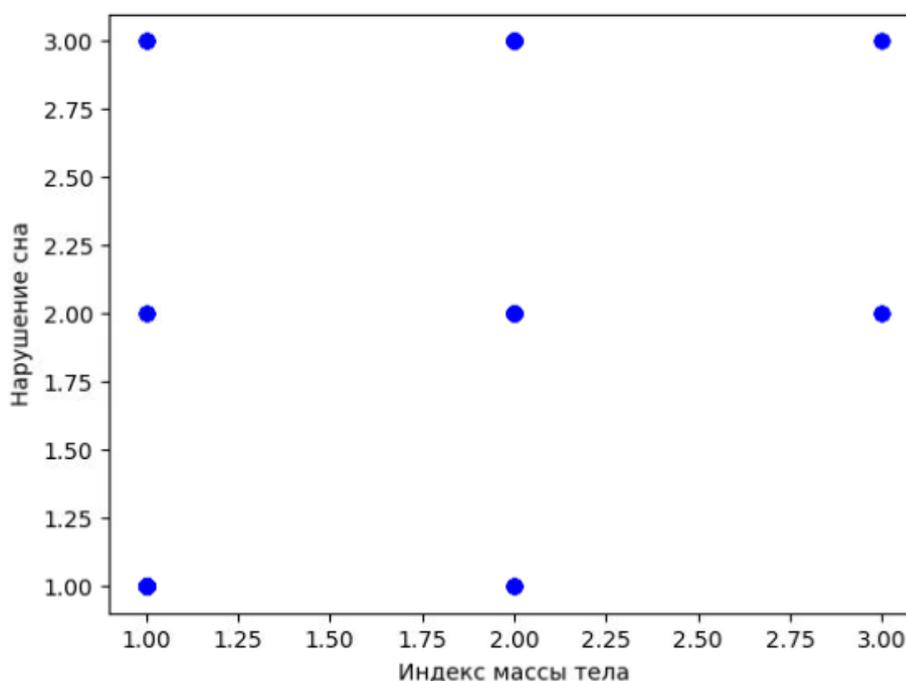


Рисунок 22 – Точечная диаграмма зависимости нарушения сна от ИМТ

С помощью библиотеки Sweetviz разработан аналитический отчет (рисунок 23).

```
import sweetviz as sv
report = sv.analyze(ds)
report.show_html("Med_rep.html")
```

Рисунок 23 – Код формирования отчета

Sweetviz – это библиотека Python с открытым исходным кодом, которая генерирует красивые визуализации высокой плотности для запуска EDA (исследовательский анализ данных) с помощью всего двух строк кода. Результатом является полностью автономное HTML-приложение.

Фрагмент отчета показан на рисунке 24.

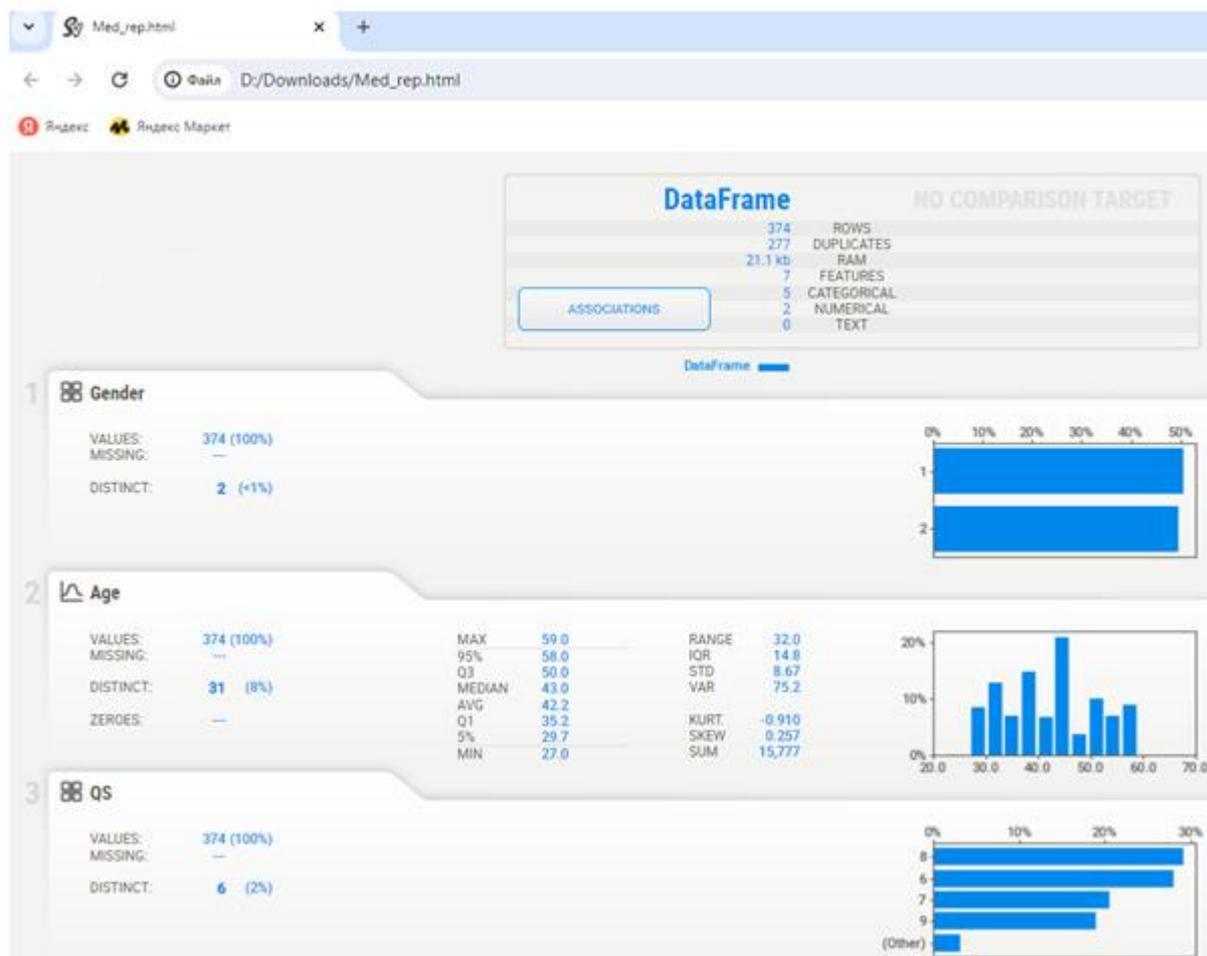


Рисунок 24 – Фрагмент аналитического отчета

Таким образом, разработанное ПО информационной системы для анализа медицинских данных обеспечивает решение задач анализа медицинских данных.

Выводы по главе 3

Результаты функционального тестирования подтвердили работоспособность разработанного ПО информационной системы для анализа медицинских данных.

Заключение

Бакалаврская работа посвящена актуальной проблеме разработки программного обеспечения информационной системы для анализа медицинских данных.

Актуальность выпускной квалификационной работы обусловлена необходимостью разработки ПО информационной системы для анализа медицинских данных, обеспечивающего высокую эффективность проведенного анализа.

В результате выполнения бакалаврской работы были решены следующие задачи:

- выполнена постановка задачи на разработку ПО информационной системы для анализа медицинских данных. По своим функциональным и архитектурным особенностям ПО информационной системы для анализа медицинских данных относится к категории ПО МИАС. ПО МИАС должно обеспечивать возможности исследования и визуализации медицинских данных, которые могут способствовать процессу лучшего понимания полученных результатов. ПО МИАС должно обеспечивать интеллектуальный анализ данных (Data mining), способный обрабатывать большие объемы данных с помощью технологий МО. Сформулированным требованиям к ПО МИАС не соответствует ни одна из представленных программ. Главный недостаток рассмотренного ПО заключается в сложности проведения анализа и избыточной функциональности. Поэтому представляет интерес разработка специального ПО МИАС, предназначенного для анализа медицинских данных;
- с помощью методологии RUP спроектировано ПО информационной системы для анализа медицинских данных. Главным преимуществом методологии RUP для рассматриваемого случая является

возможность применения для небольших проектов. На начальной фазе проектирования RUP определяются основные требования, ограничения и ключевая функциональность продукта. Создается базовая версия модели прецедентов. На фазе уточнения RUP создается архитектура ПО. Архитектура ПО служит основой для всего проекта разработки, позволяя руководству проекта экстраполировать необходимые задачи для участвующих команд;

- реализовано и протестировано ПО информационной системы для анализа медицинских данных. Для реализации ПО выбраны язык Python и среда разработки Replit. Для формирования аналитического отчета используется библиотека Sweetviz;
- диаграммы классов помогают анализировать, проектировать и документировать программные системы. Для тестирования разработанного ПО использован метод функционального тестирования продукта.

Результаты тестирования подтвердили работоспособность разработанного ПО информационной системы для анализа медицинских данных.

Список используемой литературы и используемых источников

1. Аналитическая платформа Loginom [Электронный ресурс]. URL: <https://loginom.ru/platform> (дата обращения: 10.04.2024).
2. Виноградова М. В., Белоусова В.И. Унифицированный процесс разработки программного обеспечения : учебное пособие. М: МГТУ им. Н.Э. Баумана, 2015. 82 с. URL: <https://www.iprbookshop.ru/134990.html> (дата обращения: 10.04.2024).
3. Десять библиотек Python для автоматического разведочного анализа данных. URL: <https://vc.ru/u/1389654-machine-learning/758486-10-bibliotek-python-dlya-avtomaticheskogo-razvedochnogo-analiza-dannyh>(дата обращения: 06.04.2024).
4. Доказательная медицина для всех [Электронный ресурс]. URL: <https://medspecial.ru/> (дата обращения: 15.04.2024).
5. Карпов О.Э., Никуличев А.А., Пензин О.В., Субботин С.А., Андриков Д.А., Перфильев А.Е. Архитектура медицинских информационных систем нового поколения // Вестник Национального медико-хирургического Центра им. Н.И. Пирогова. 2019. Т. 14. № 3. С.126-134.
6. Коцюба И.Ю., Чунаев А.В., Шиков А.Н. Методы оценки и измерения характеристик информационных систем. Учебное пособие. СПб: Университет ИТМО, 2015. 264 с.
7. Леоненков А. В. Объектно-ориентированный анализ и проектирование с использованием UML и IBM Rational Rose : учебное пособие. М. : ИНТУИТ, Ай Пи Ар Медиа, 2020. 317 с. [Электронный ресурс]. URL: <https://www.iprbookshop.ru/97554.html> (дата обращения: 10.02.2024).
8. Маккинли У. Python и анализ данных. Саратов : Профобразование, 2019. 482 с. URL: <https://www.iprbookshop.ru/88752.html> (дата обращения: 06.04.2024).

9. Платформа RapidMiner [Электронный ресурс]. URL: <https://my.rapidminer.com/nexus/account/index.html#home> (дата обращения: 10.04.2024).

10. Подходы к разработке ПО: как правильно выбрать методологию разработки программного обеспечения [Электронный ресурс]. URL: <https://issoft.by/blog/podkhody-k-razrabotke-po-kak-pravilno/> (дата обращения: 10.04.2024).

11. Самуйлов С. В. Объектно-ориентированное моделирование на основе UML : учебное пособие. Саратов : Вузовское образование, 2016. 37 с. URL: <https://www.iprbookshop.ru/47277.html> (дата обращения: 15.04.2024).

12. Функциональное тестирование ПО [Электронный ресурс]. URL: <https://gb.ru/blog/funktsionalnoe-testirovanie-po/> (дата обращения: 10.04.2024).

13. Knime Analytics Platform [Электронный ресурс]. URL: <https://www.knime.com/>

14. Python IDLE [Электронный ресурс]: Документация. URL: <https://docs.python.org/3/library/idle.html> (дата обращения: 10.04.2024).

15. Replit [Электронный ресурс]: Описание среды. URL: <https://replit.com/~> (дата обращения: 10.04.2024).

16. Santos-Pereira J., Gruenwald L., Bernardino J. Top data mining tools for the healthcare industry, Journal of King Saud University - Computer and Information Sciences, Volume 34, Issue 8, Part A, 2022, P. 4968-4982.

17. Scikit-learn 1.4.1.post1 [Электронный ресурс]: Описание проекта. URL: <https://pypi.org/project/scikit-learn/> (дата обращения: 10.04.2024).

18. Scrum: методология гибкой разработки [Электронный ресурс]. URL: <https://gb.ru/blog/skrum/> (дата обращения: 10.04.2024).

19. Sleep Health and Lifestyle Dataset [Электронный ресурс]. URL: <https://www.kaggle.com/datasets/uom190346a/sleep-health-and-lifestyle-dataset> (дата обращения: 15.04.2024).

20. Software Requirements [Электронный ресурс]. URL: <http://beerpvolume.com/oop/2020/software-requirements/> (дата обращения: 10.02.2024).

21. UML 2.ru – Сообщество Аналитиков [Электронный ресурс]. URL: <https://www.uml2.ru/> (дата обращения: 10.04.2024).

22. Unified Modeling Language (UML) Diagrams [Электронный ресурс]. URL: <https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/> (дата обращения: 10.04.2024).

23. Visual Paradigm [Электронный ресурс]. URL: <https://online.visual-paradigm.com/> (дата обращения: 10.02.2024).

24. Waterfall Methodology: A Comprehensive Guide [Электронный ресурс]. URL: <https://www.atlassian.com/agile/project-management/waterfall-methodology> (дата обращения: 10.04.2024).