

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Национальный исследовательский ядерный университет «МИФИ»

Балаковский инженерно-технологический институт –

филиал федерального государственного автономного образовательного учреждения высшего образования
«Национальный исследовательский ядерный университет «МИФИ»

ФАКУЛЬТЕТ атомной энергетики и технологий

КАФЕДРА «Информационные системы и технологии»

На правах рукописи

УДК 004.4

СЕМЕНОВ МАКСИМ АЛЕКСАНДРОВИЧ

РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ВЗАИМОДЕЙСТВИЯ С КЛИЕНТАМИ В МЕДИЦИНСКОМ ЦЕНТРЕ

Выпускная квалификационная работа бакалавра

Направление подготовки 09.03.02

(код, наименование)

«Информационные системы и технологии»

Выпускная квалификационная
работа защищена

« » 2025 г.

Оценка

Секретарь ГЭК /Н.М. Виштак/

Балаково 2025

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный исследовательский ядерный университет «МИФИ»
Балаковский инженерно-технологический институт –
филиал федерального государственного автономного образовательного учреждения высшего образования
«Национальный исследовательский ядерный университет «МИФИ»

Факультет

АЭТ

Кафедра Информационные системы

и технологии

Направление подготовки 09.03.02

«Информационные системы

и технологии»

Группа ИФСТ-5з

УТВЕРЖДАЮ

Зав. кафедрой ИСТ

(подпись)

Очкур Г.В.

(фамилия, имя, отчество)

«11» апреля 2025 г.

ЗАДАНИЕ НА БАКАЛАВРСКУЮ РАБОТУ

(выпускную квалификационную работу)

1. Фамилия, имя, отчество студента Семенов Максим Александрович

2. Тема работы Разработка программного модуля взаимодействия с клиентами
в медицинском центре

3. Срок сдачи студентом готовой работы 11.06.2025 г.

4. Место выполнения работы БИТИ НИЯУ МИФИ

5. Руководитель работы (проекта) д.п.н., к.т.н., декан ФПКПП, профессор
кафедры ИСТ Виштак Ольга Васильевна

(фамилия, имя, отчество, должность, место работы)

6. Консультанты работы Очкур Г.В., заведующий кафедрой «Информацион-
ные системы и технологии», Михеев И.В., старший преподаватель кафедры «Ин-
формационные системы и технологии»

(фамилия, имя, отчество, должность, место работы)

7. Цель работы: Разработка программного модуля взаимодействия с клиентами в медицинском центре (ПМ ВКМЦ)

8. Задание:

а) литература и обзор работ, связанных с работой _____

1. Хассан Гома. UML. Проектирование систем реального времени, распределенных и параллельных приложений / Хассан Гома. – М.: ДМК-Пресс, 2024. – 700 с.

2. Теория информационных процессов и систем 2-е изд., пер. и доп. Учебник и практикум для академического бакалавриата / В.Н. Волкова – М: Юрайт, 2024. – 433с.

3. Крокфорд Д. JavaScript. Сильные стороны; Питер - М., 2023. - 262 с.

б) расчетно-конструкторская, теоретическая, технологическая часть ____

1. Анализ предметной области

2. Классификация готовых программных продуктов ПМ ВКМЦ

3. Планирование и определение затрат на разработку ПМ ВКМЦ

4. Разработка технического задания ПМ ВКМЦ

в) экспериментальная часть _____

1. Разработка объектно-ориентированной модели ПМ ВКМЦ

2. Проектирование навигационной структуры ПМ ВКМЦ

3. Выбор инструментальных средств создания ПМ ВКМЦ

4. Описание интерфейса ПМ ВКМЦ

5. Тестирование работы ПМ ВКМЦ

9. Отчетный материал работы:

а) пояснительная записка;

б) графический материал (с указанием обязательных чертежей).

10. Консультант по работе (с указанием относящихся к ним разделов работы)

Раздел	Консультант	Подпись, дата	
		Задание выдал	Задание принял
Проектирование программного модуля администрирования	Очкур Г.В.	11.04.2025г.	11.06.2025г.
Реализация программного модуля ведения учетных записей медицинского центра	Михеев И.В.	11.04.2025г.	11.06.2025г.

Календарный план работы над выпускной квалификационной работой

(составляется руководителем с участием студента в течение первой недели с начала
преддипломной практики)

№ п\п	Наименование этапов работы	Сроки выполнения этапов	Степень готовности ВКР в % к объему работы	Отметка о выполнении
1	Введение Анализ требований для раз- работки программного мо- дуля Анализ предметной области. Классификация готовых программных модулей. Разработка технического задания	11.04.25г.- 19.04.25г.	30%	
2	Проектирование программ- ного модуля Разработка объектно- ориентированной модели Проектирование навига- ционной структуры ресурса. Обзор и выбор инструмен- тальных средств создания программного модуля	21.04.25г.- 26.04.25г.	60%	
3	Реализация программного модуля Описание интерфейса про- граммного модуля Тестирование программного модуля Заключение	28.04.25г.- 25.05.25г.	90%	
4	Оформление презентации. Сдача ВКР.	05.06.25г.	100%	

Дата выдачи задания - 11.04.2025г.

Руководитель выпускной
квалификационной
работы

(подпись)

(фамилия имя, отчество)

Задание принял к исполнению

(подпись)

(фамилия, имя, отчество)

«11» апреля 2025 г.

СОДЕРЖАНИЕ

Введение	6
1 Анализ требований для разработки программного модуля ведения учетных записей медицинского центра	8
1.1 Анализ задач, функций и требований программного модуля ведения учетных записей медицинского центра	9
1.2 Обзор существующих методов и программных разработок в создании информационных систем медицинского назначения	20
1.3 Планирование и определение затрат на разработку программного модуля ведения учетных записей медицинского центра	22
1.4 Техническое задание на разработку программного модуля ведения учетных записей медицинского центра	29
2 Проектирование программного модуля администрирования	36
2.1 Проектирование программного модуля ведения учетных записей медицинского центра с использованием объектного моделирования	36
2.2 Выбор языка и среды разработки программного модуля ведения учетных записей медицинского центра	41
3. Реализация программного модуля ведения учетных записей медицинского центра.	46
3.1. Реализация программного модуля ведения учетных записей медицинского центра.	46
3.2. Описание интерфейса программного модуля ведения учетных записей медицинского центра	56
3.3 Тестирование программного модуля ведения учетных записей медицинского центра	60
Заключение	66
Список использованных источников	67

					ИФСТ. 466452.008 ПЗ		
Изм.	Лист	№ докум.	Подпись	Дата			
Разраб.	Семенов				<i>Разработка программного модуля взаимодействия с клиентами в медицинском центре. Пояснительная записка</i>	Лит.	Лист
Рук. ВКР	Виштак						5
Консульт.	Михеев						68
Н.контр.	Виштак					БИТИ ИФСТ-5з	
Утв.	Очкур						

ВВЕДЕНИЕ

Современные медицинские учреждения являются сложными организациями, в которых эффективность работы напрямую зависит от уровня автоматизации процессов.

В настоящее время медицинские учреждения проходят процесс внедрения информационных технологий для адаптации к современным требованиям.

Процесс информатизации охватывает различные аспекты здравоохранения, начиная от внедрения медицинских информационных систем (МИС) до использования телемедицинских технологий.

Важным аспектом является возможность стабильно выполнять предоставляемые услуги, для этого необходимо разрабатывать и внедрять программные модули, которые позволяют сотрудникам быстро и оперативно сообщать об ошибках и требованиях, без которых эффективность выполнения процессов уменьшается.

В условиях динамично развивающихся информационных технологий и растущего объёма данных, ручной подход к управлению данными становится неэффективным и трудозатратным.

Важнейшим направлением совершенствования являются:

- применение современных технологических процессов;
- совершенствование методов организации и управления;
- эффективное использование трудовых ресурсов;
- улучшение условий труда сотрудников;
- адаптация к новым технологиям.

Для решения этих задач актуальным является внедрение информационных технологий и систем, которые позволяют выстраивать наилучшие взаимоотношения с клиентами.

Целью выпускной квалификационной работы является разработка программного модуля взаимодействия с клиентами в медицинском центре,

которая позволит повысить эффективность работы путем автоматизации процессов.

Для достижения цели необходимо выполнить следующие задачи:

- провести анализ предметной области;
- провести классификацию программных модулей обратной связи;
- разработать техническое задание;
- провести планирование и определение затрат на разработку ПМ;
- разработать объектно-ориентированную модель;
- спроектировать навигационную структуру системы;
- провести анализ и выбрать инструментальные средства для разработки системы;
- разработать программный модуль;
- провести тестирование программного модуля.

Теоретическая значимость выпускной квалификационной работы заключается во всестороннем анализе концептуальных основ и средств разработки информационных ресурсов организации.

Практическая значимость полученных результатов в разработке программного модуля взаимодействия с клиентами в медицинском центре заключается в повышении эффективности, упрощения процесса взаимодействия с клиентами, создание централизованного хранилища с обращениями клиентов.

1 АНАЛИЗ ТРЕБОВАНИЙ ДЛЯ РАЗРАБОТКИ ПРОГРАММНОГО МОДУЛЯ ВЕДЕНИЯ УЧЕТНЫХ ЗАПИСЕЙ МЕДИЦИНСКОГО ЦЕНТРА

Специализацией компании «Информационно-медицинский центр» (ИМЦ) является информатизация здравоохранения, включая комплексные решения для медицинских организаций, органов управления здравоохранением и территориальных фондов ОМС.

С момента организации предприятие стремительно развивается, специализируясь на разработке программных продуктов, которые являются комплексными решениями для медицинских организаций, органов управления здравоохранением и территориальных фондов ОМС.

ООО «ИМЦ» имеет возможность адаптировать систему под нужды организаций, их эволюционное развитие в зависимости от совершенствования технической инфраструктуры учреждения здравоохранения [1].

На рисунке 1 представлена организационная структура ООО «ИМЦ».

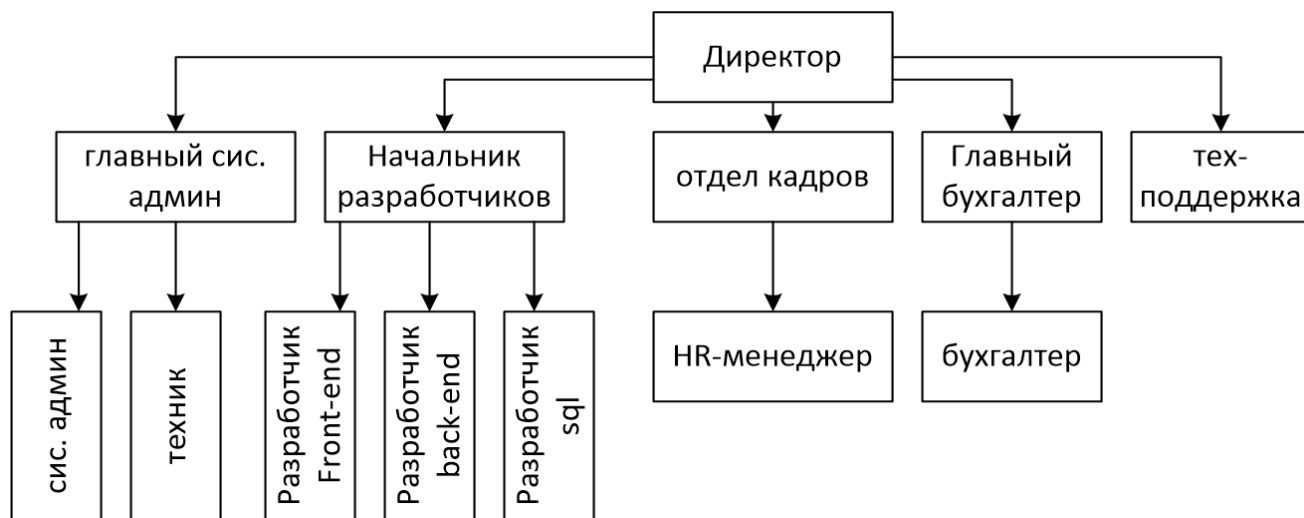


Рисунок 1 - Организационная структура ООО «ИМЦ»

Управление предприятием ООО «ИМЦ» осуществляется директором, он является руководителем предприятия. В подчинении у директора находятся все начальники IT отделов и бухгалтера. Главный бухгалтер ведёт отчеты по всему

предприятию. Начальник IT отделов следит за работой своей задачи и распределяет задачи.

Главный системный администратор распределяет обязанности между сотрудниками своего отдела.

Системный администратор обеспечивает штатную работу компьютерной техники, сети и программного обеспечения.

Техник занимается обслуживанием, профилактикой и ремонтом различного типа оборудования.

Начальник разработчиков занимается реализацией одного либо нескольких проектов.

Разработчик front-end разрабатывает визуальную часть веб-сайта.

Разработчик back-end разрабатывает логику продукта.

Разработчик sql разрабатывает базу данных.

Техническая поддержка - отдел, обрабатывающий обращения клиентов.

HR-менеджер организует управление персоналом в компании.

Предприятие ООО «ИМЦ» занимается разработкой компьютерного программного обеспечения и его поддержкой, соответственно предприятие нуждается в постоянной возможности быстрого реагирования на существующие ошибки своего продукта от клиентов.

1.1 Анализ задач, функций и требований программного модуля ведения учетных записей медицинского центра

Целью деятельности предприятия ООО «ИМЦ» является разработка компьютерного программного обеспечения для выполнения требований.

Задачами предприятия ООО «ИМЦ» являются:

- получение прибыли предприятия ООО «ИМЦ»;
- обеспечение потребителей программным обеспечением в соответствии с договорами;

- обеспечение персонала предприятия заработной платой, нормальными условиями труда и возможностью профессионального роста;
- создание рабочих мест для населения в пределах муниципального округа.

К основному виду деятельности ООО «ИМЦ» относится разработка компьютерного программного обеспечения.

К дополнительным видам деятельности ООО «ИМЦ» относятся:

- торговля оптовая компьютерами, периферийными устройствами к компьютерам и программным обеспечением;
- торговля оптовая неспециализированная;
- торговля розничная компьютерами, периферийными устройствами к ним и программным обеспечением в специализированных магазинах;
- деятельность консультативная и работы в области компьютерных технологий;
- деятельность, связанная с использованием вычислительной техники и информационных технологий, прочая;
- деятельность по обработке данных, предоставление услуг по размещению информации и связанная с этим деятельность;
- деятельность по созданию и использованию баз данных и информационных ресурсов;
- деятельность по оказанию консультационных и информационных услуг;
- научные исследования и разработки в области естественных и технических наук прочие;
- деятельность по предоставлению прочих вспомогательных услуг для бизнеса, не включенная в другие группировки;
- ремонт компьютеров и периферийного компьютерного оборудования.

Важным процессом организации ООО «ИМЦ» является обработка обращений клиентов. Благодаря этому процессу клиенты могут оставлять свои тре-

бования, пожелания и информировать о найденных ошибках программного продукта.

Для более наглядного представления процесса обработки обращения клиентов составлена функциональная блок-схема, которая изображена на рисунке 2.

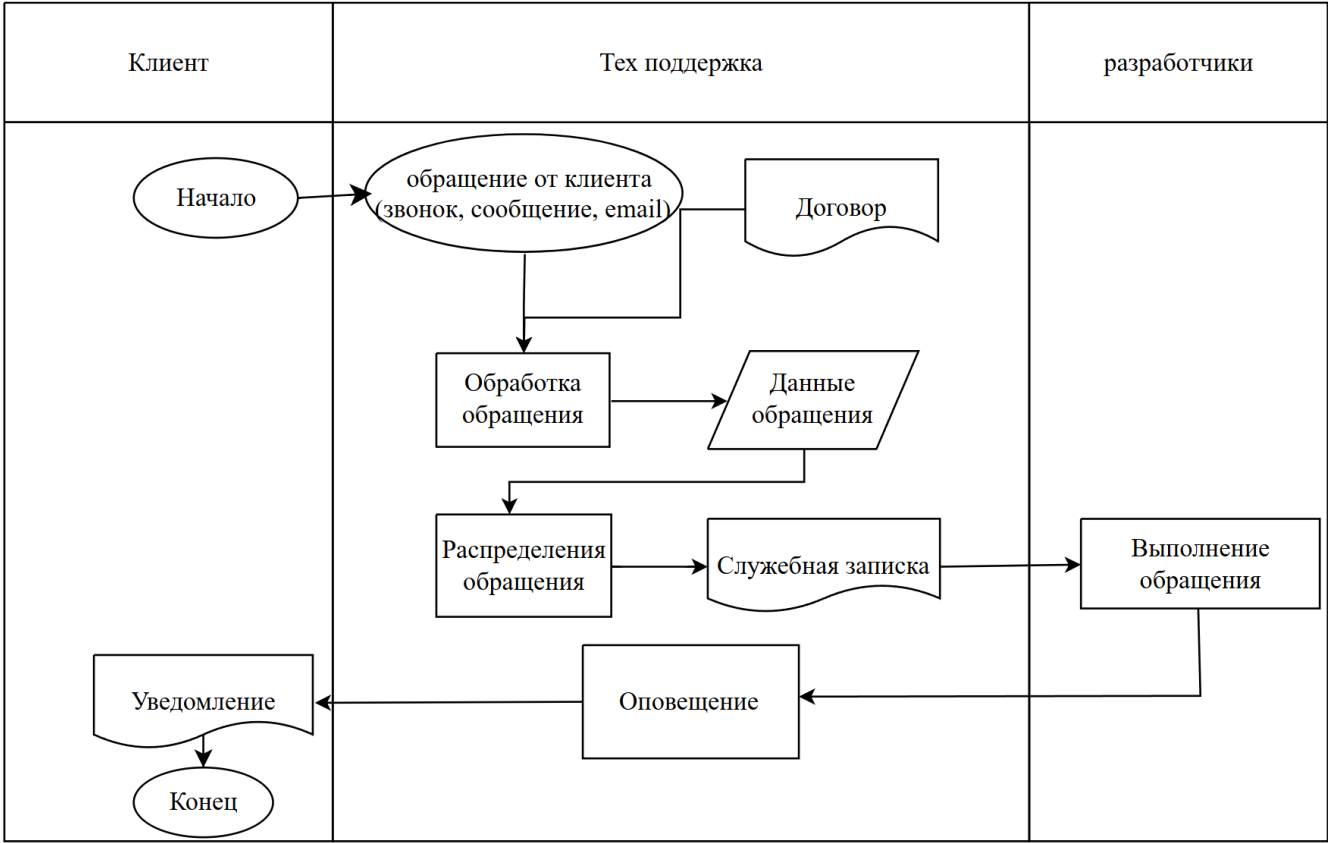


Рисунок 2 - Функциональная блок-схема процесса обработки обращения клиентов

Основными документами, регламентирующими предпринимательскую деятельность ООО «ИМЦ» являются:

- Конституция РФ;
- Федеральный закон от 08.02.1998 N 14-ФЗ (ред. от 23.04.2018) «Об обществах с ограниченной ответственностью»;
- Лицензия ФСТЭК на деятельность по технической защите конфиденциальной информации;
- Лицензия ФСБ на осуществление деятельности по разработке, производству, распространению, техническому обслуживанию шифровальных (криптографических) средств.

Для разработки моделей процессов информационно-технической деятельности отдела технической поддержки ООО «ИМЦ» выбрана задача – обработка обращения клиентов.

Для разработки модели процесса обработки обращения от клиентов будет использоваться методология IDEF0.

IDEF0 - это методология функционального моделирования и графическая нотация, предназначенная для формализации и описания бизнес-процессов.

Функциональная модель IDEF0 представляет собой набор блоков, каждый из которых представляет собой черный ящик со входами и выходами, управлением и механизмами, которые детализируются (декомпозируются) до необходимого уровня [5].

Функциональный блок контекстной диаграммы процесса обработки обращения клиентов представлена на рисунке 3.

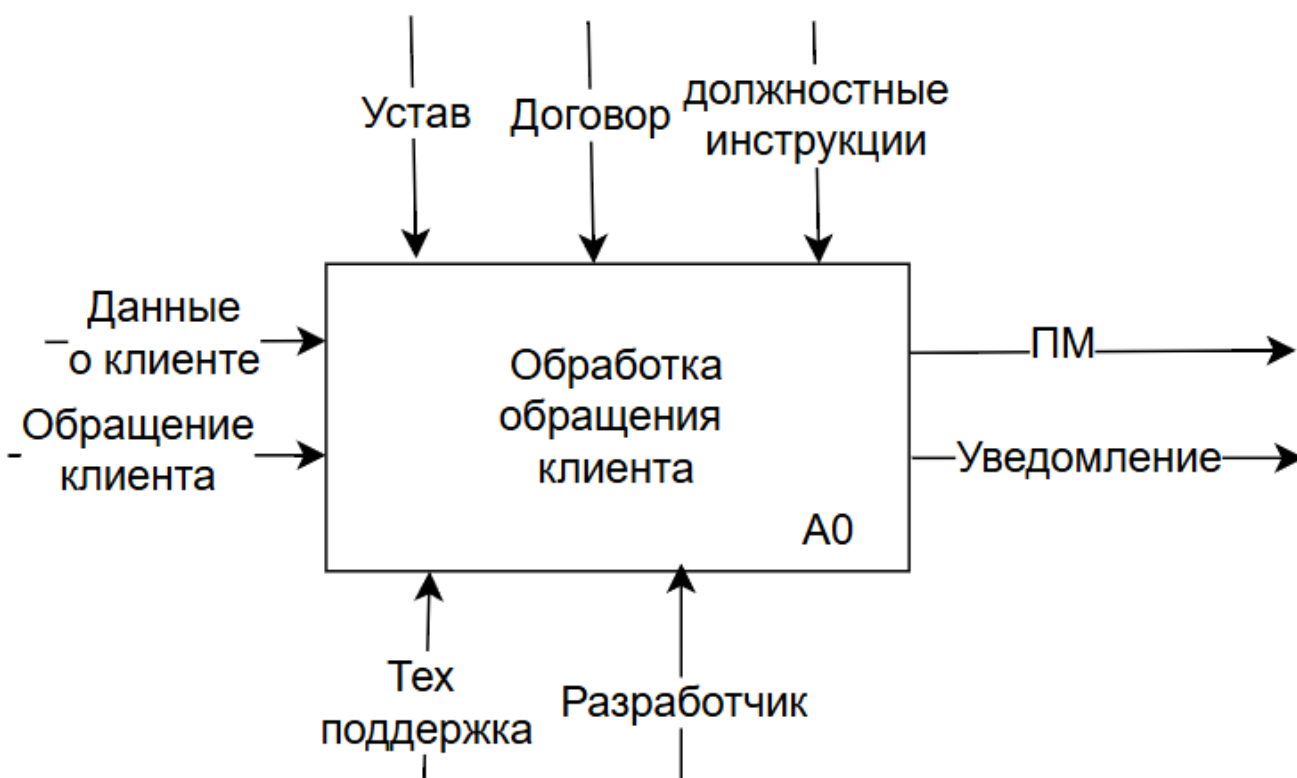


Рисунок 3 - Функциональный блок контекстной диаграммы процесса обработки обращения клиентов

Для более детального изучения процесса обработки обращения клиентов необходимо разработать декомпозицию контекстной диаграммы.

The diagram illustrates the process of processing a client's request, divided into four stages (A1, A2, A3, A4) and their interactions with external inputs and outputs.

- Stage A1: Обработка обращения (Processing of the request)**
 - Inputs: Договор (Contract), Устав (Charter), Данные о клиенте (Client data), and Обращение клиента (Client request).
 - Output: данные о обращении (Request data) to Stage A2.
- Stage A2: Распределения обращения (Distribution of the request)**
 - Inputs: должностные инструкции (Job instructions) and данные о обращении (Request data) from Stage A1.
 - Outputs: C3 to Stage A3 and T3 to Stage A3.
- Stage A3: Выполнения обращения (Execution of the request)**
 - Inputs: C3, T3, должностные инструкции (Job instructions), and Разработчик (Developer).
 - Output: Выполненное обращение (Executed request) to Stage A4.
- Stage A4: Оповещение (Notification)**
 - Inputs: Выполненное обращение (Executed request) from Stage A3 and Уведомление (Notification) from the external environment.
 - Output: ПМ (PM) to the external environment.
- External Interactions:**
 - Договор (Contract) and Устав (Charter) also have direct paths to Stage A3.
 - Тех поддержка (Technical support) has a path to Stage A4.

Клиент, обращается к технической поддержке из различных источников, передавая свои обращения и персональные данные, необходимые для исправления программного модуля, после чего сотрудник технической поддержки на основе договора проверяет наличие у клиента сопровождения технической поддержки и изучает другую информацию, необходимую для принятия обращения.

Далее происходит выполнение обращения либо разработчиком, либо сотрудником технической поддержки.

В этом процессе можно выделить проблемы:

- техническая поддержка принимает обращения от клиентов через множество разных источников: социальные сети, Email, по телефону и СМС;
- техническая поддержка обрабатывает задачи в системе;

– после выполнения обращения клиента техническая поддержка оповещает клиента о выполненной работе.

Для решения этих проблем нужно провести оптимизацию данного процесса с помощью создания программного модуля. Создание программного модуля для предприятия ООО «ИМЦ» позволит сократить нагрузку сотрудников технической поддержки, сократить трудозатраты отдела технической поддержки и увеличить эффективность работы технической поддержки и доход ООО «ИМЦ».

Для оптимизации процесса требуется создать контекстную диаграмму модели ТО-ВЕ, которая должна решить проблемы в происходящем процессе обработки обращения клиентов.

Контекстная диаграмма процесса обработки обращения клиентов с использованием ПМ представлена на рисунке 5.

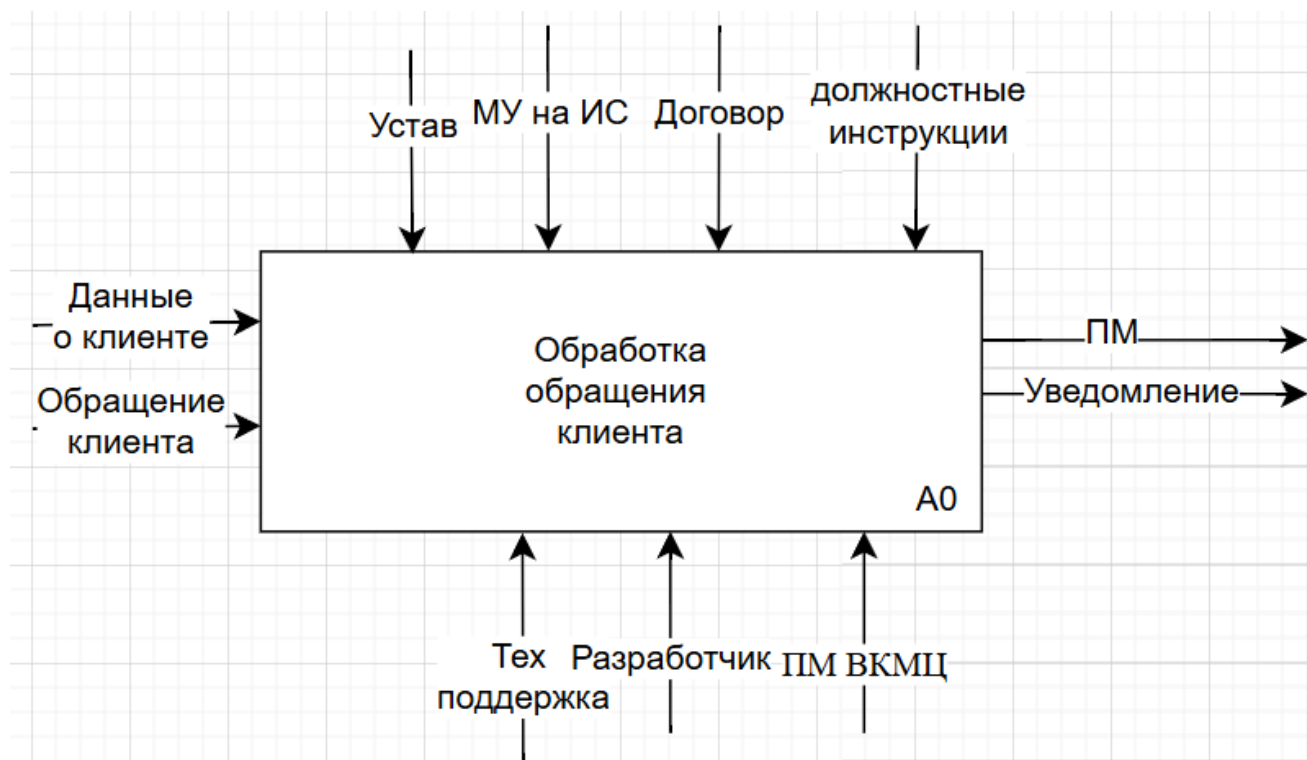


Рисунок 5 - Контекстная диаграмма процесса обработки обращения клиентов с использованием ПМ

Далее необходимо разработать декомпозицию контекстной диаграммы для модели ТО-ВЕ процесса обработки обращения клиентов.

Декомпозиция контекстной диаграммы для модели ТО-ВЕ, процесса обработки обращения клиентов представлена на рисунке 6.

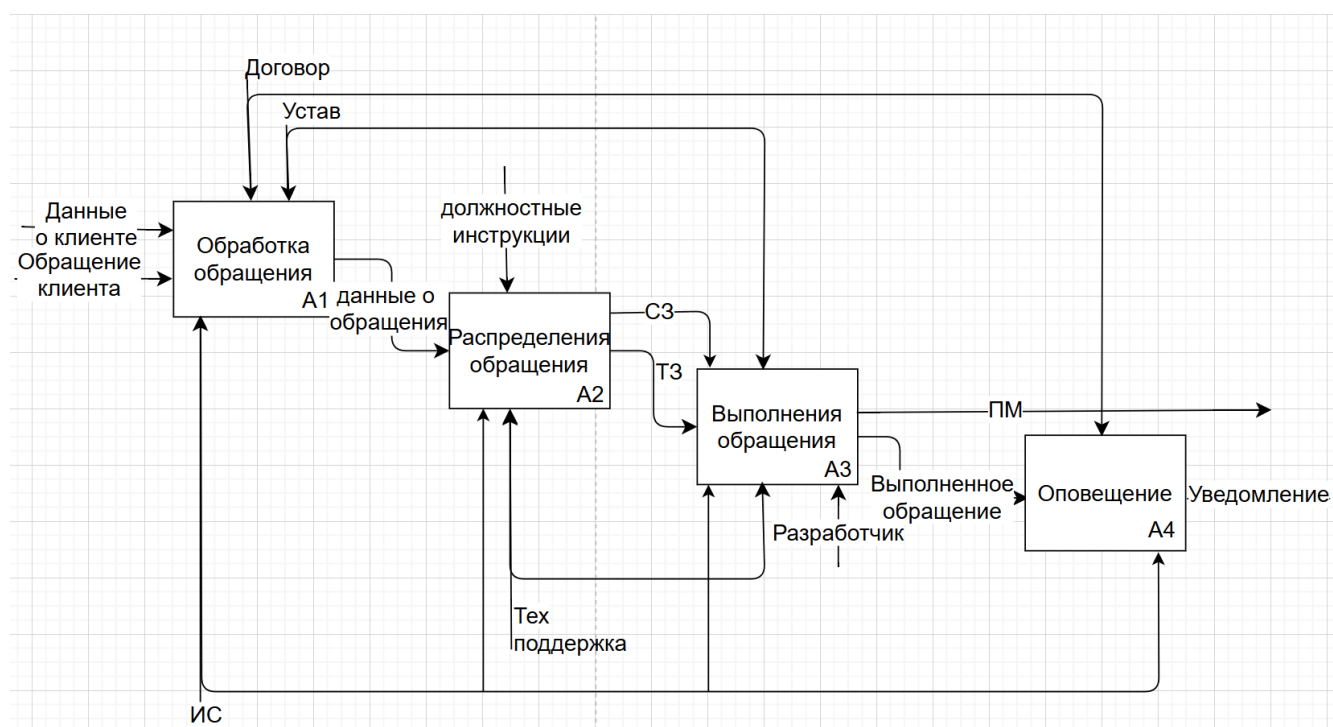


Рисунок 6 – Декомпозиция контекстной диаграммы процесса обработки обращения клиентов с использованием ПМ

Клиент проходит процесс авторизации в программном модуле, в этот момент происходит проверка клиента на обслуживание технической поддержки, после чего клиенту предоставляется доступ к ПМ, далее происходит заполнение формы обращения, затем техническая поддержка осуществляет процесс определения исполнителя обращения. После чего исполнитель получает уведомление об обращении и приступает к его выполнению. После выполнения обращения исполнитель изменяет статус обращения «на проверку», далее ПМ уведомляет клиента о проделанных работах.

Данное изменение процесса приводит к созданию одного источника получения данных от клиента и уменьшает нагрузку отдела технической поддержки за счет автоматизации процесса посредством ПМ.

DFD - диаграммы потоков данных, методология графического структурного анализа, описывающая внешние по отношению к системе источники и адреса-

ты данных, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ [2].

На рисунке 7 представлено диаграмма потоков данных процесса обработки обращения клиентов.

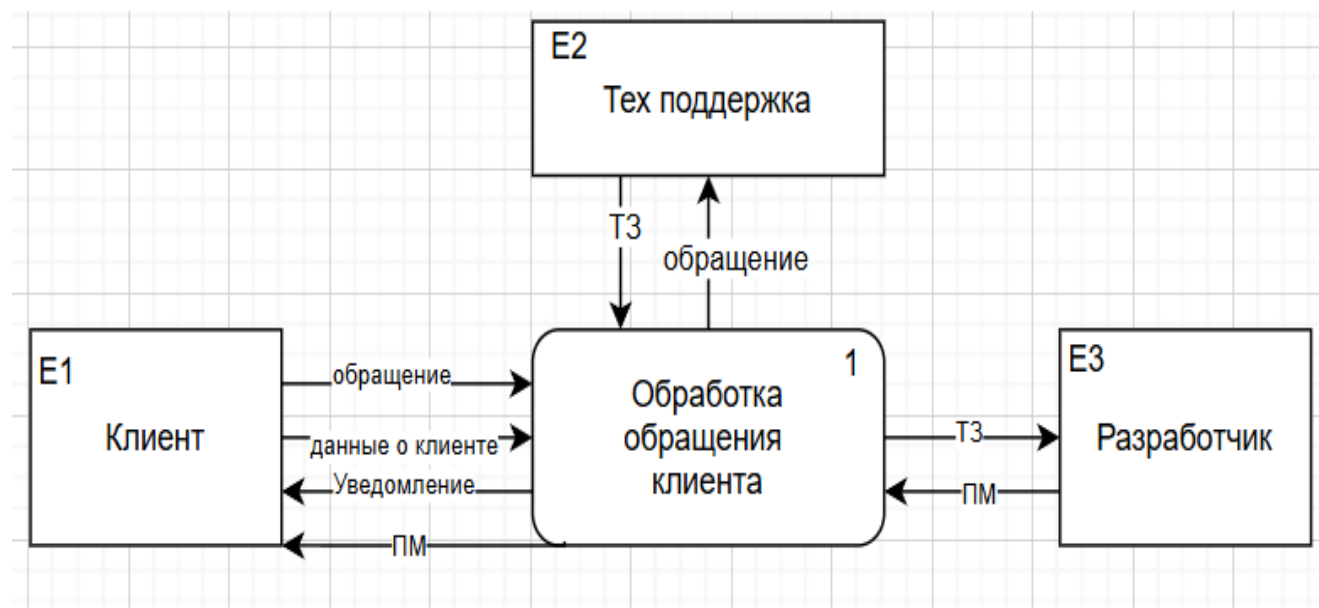


Рисунок 7 - Контекстная диаграмма потоков данных процесса обработки обращения клиентов

На этой диаграмме выделены 3 сущности: клиент, разработчик, техническая поддержка. В процессе обработки обращения клиентов клиент передает свое обращение и данные на изменение программного продукта из разных источников по email, по телефону, по мессенджерам, по сообщениям телефона.

Данное обращение принимает техническая поддержка и производит обработку обращения и распределяет обращения по сотрудникам, которые ответственны за появившуюся ошибку или появление нового функционала, после этого тех поддержка представляет конкретному исполнителю техническое задание. Далее исполнитель изучает техническое задание, на основе которого он разрабатывает или исправляет ошибку в программном модуле, после чего уведомляет техническую поддержку о выполнении технического задания.

Техническая поддержка проверяет соответствие проделанных работ с техническим заданием и уведомляет клиента о выполненных работах.

Декомпозиция процесса обработки обращения клиентов приведена на рисунке 8.

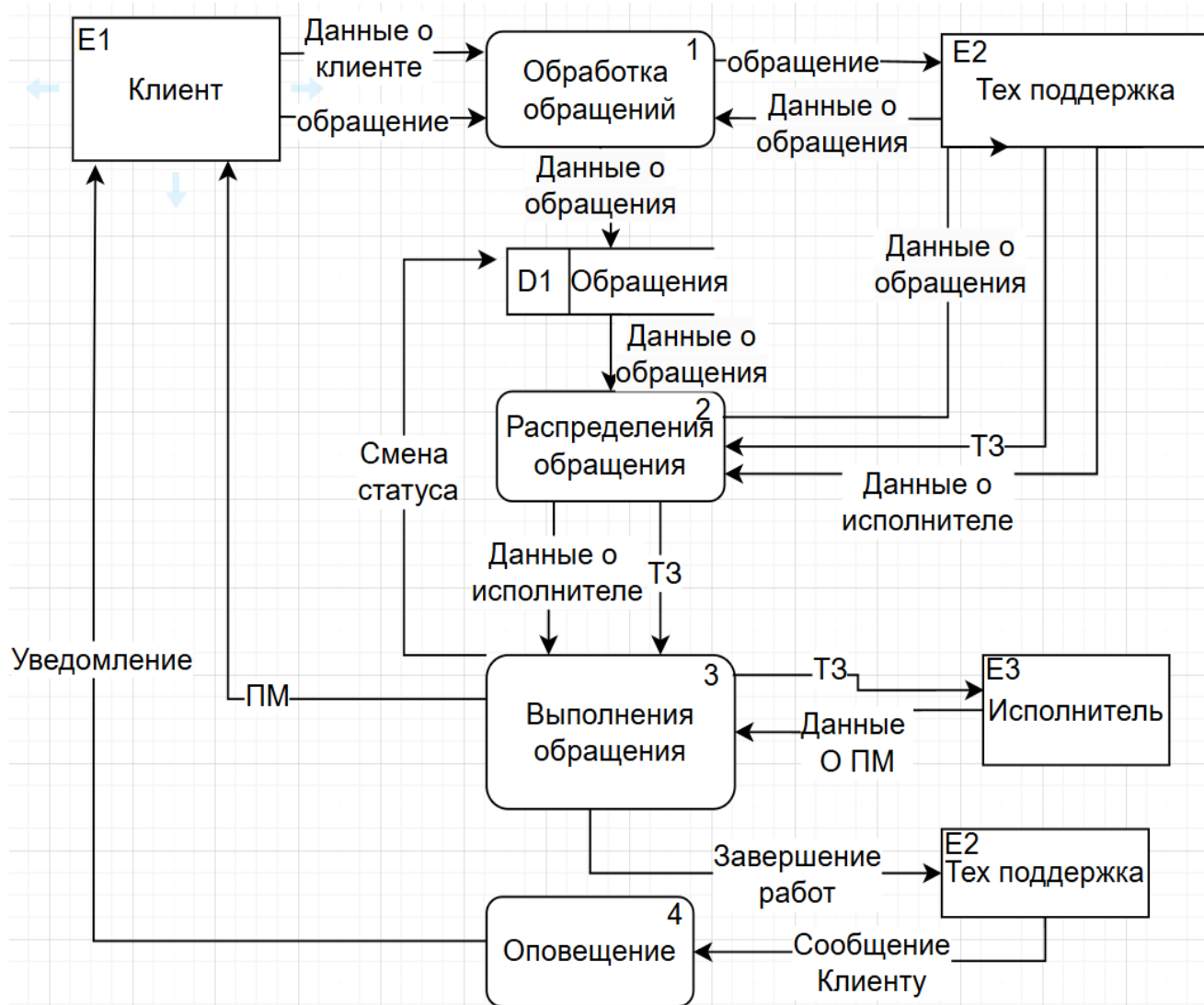


Рисунок 8 - Декомпозиция процесса обработки обращения клиентов

После внедрения программного модуля в процесс обработки обращения клиентов клиент передает свое обращение на изменение или исправление ошибки программного продукта и данные для авторизации в программный модуль, который обрабатывает эти данные и передает их в техническую поддержку, далее происходит распределение обращения.

После разработчику приходит техническое задание на основе которого он разрабатывает новый функционал или исправляет ошибку, которые передаются клиенту и программный модуль уведомляет о проделанных работ.

На рисунке 9 представлена диаграмма потоков данных процесса обработки обращения клиента с ПМ.

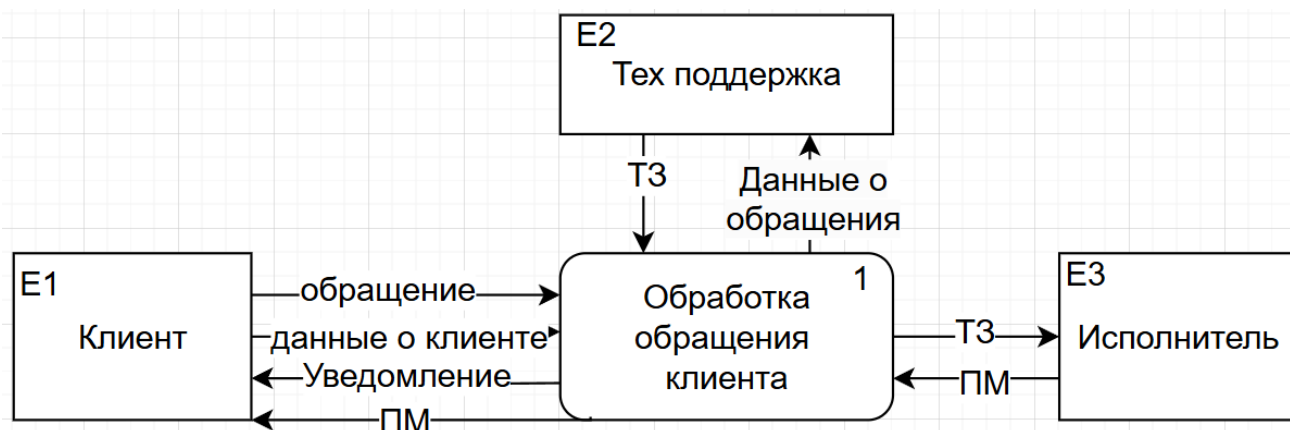


Рисунок 9 - Диаграмма потоков данных процесса обработки обращения клиента с ПМ

На рисунке 10 представлена декомпозиция диаграммы потоков данных процесса обработки обращения клиентов с ПМ.

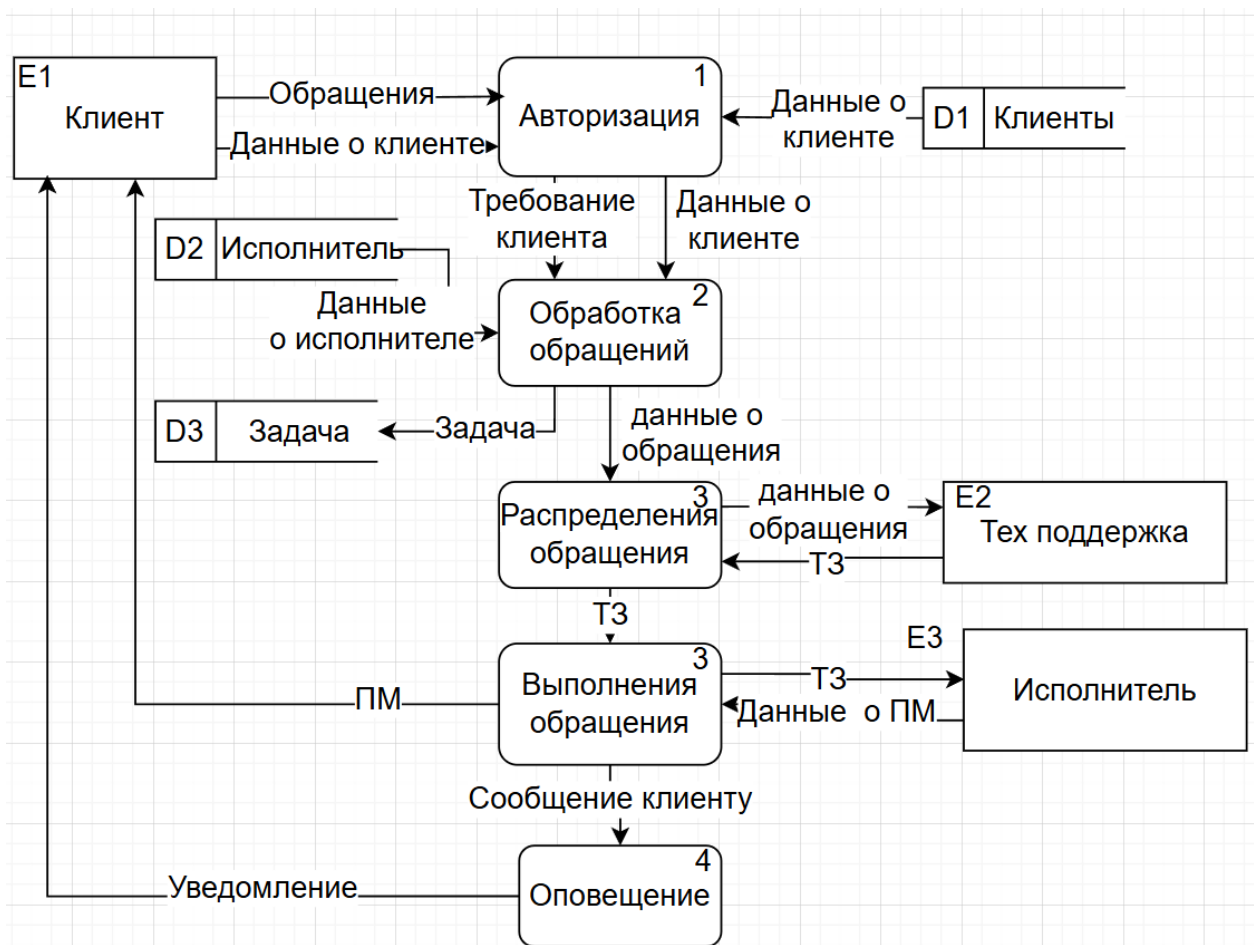


Рисунок 10 - Декомпозиция диаграммы потоков данных процесса обработки обращения клиентов с ПМ

IDEF3 — методология моделирования и стандарт документирования процессов, происходящих в системе. Метод документирования технологических процессов представляет собой механизм документирования и сбора информации о процессах. IDEF3 показывает причинно-следственные связи между ситуациями и событиями в понятной эксперту форме, используя структурный метод выражения знаний о том, как функционирует система, процесс или предприятие. [4]

На рисунке представлено 11 контекстная диаграмма процесса обработки обращения клиента.

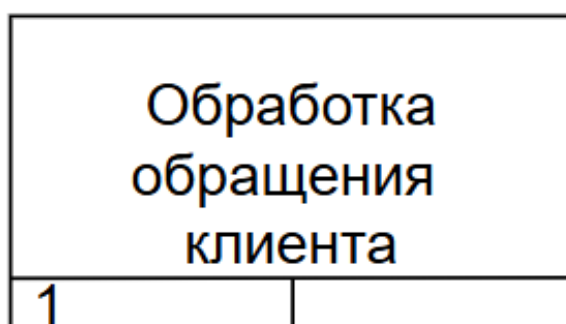


Рисунок 11 - Диаграмма процесса обработки обращения клиента

На рисунке 12 представлена декомпозиция процесса обработки обращения клиентов с стороны клиента.

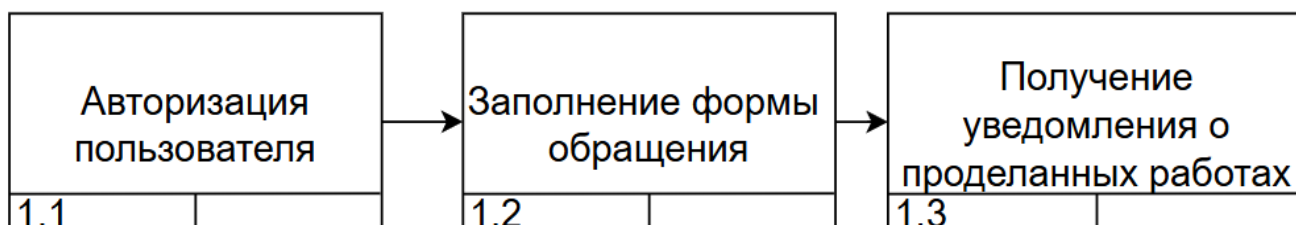


Рисунок 12 - Декомпозиция процесса обработки обращения клиентов

Клиент заходит в ПМ, вводит свой логин и пароль, после чего происходит авторизация, которая загружает данные о клиенте, после чего клиенту требуется заполнить форму с обращением в ПМ и после выполнения обращения клиента отправится уведомление клиенту о проделанных работах.

Логическая модель данных — это расширение концептуальной модели данных. Она включает в себя все сущности, атрибуты, ключи и взаимосвязи, которые представляют информацию и определяют правила [3].

Средства моделирования IDEF1X специально разработаны для построения реляционных ПМ.

На рисунке 13 представлена физическая модель ПМ.

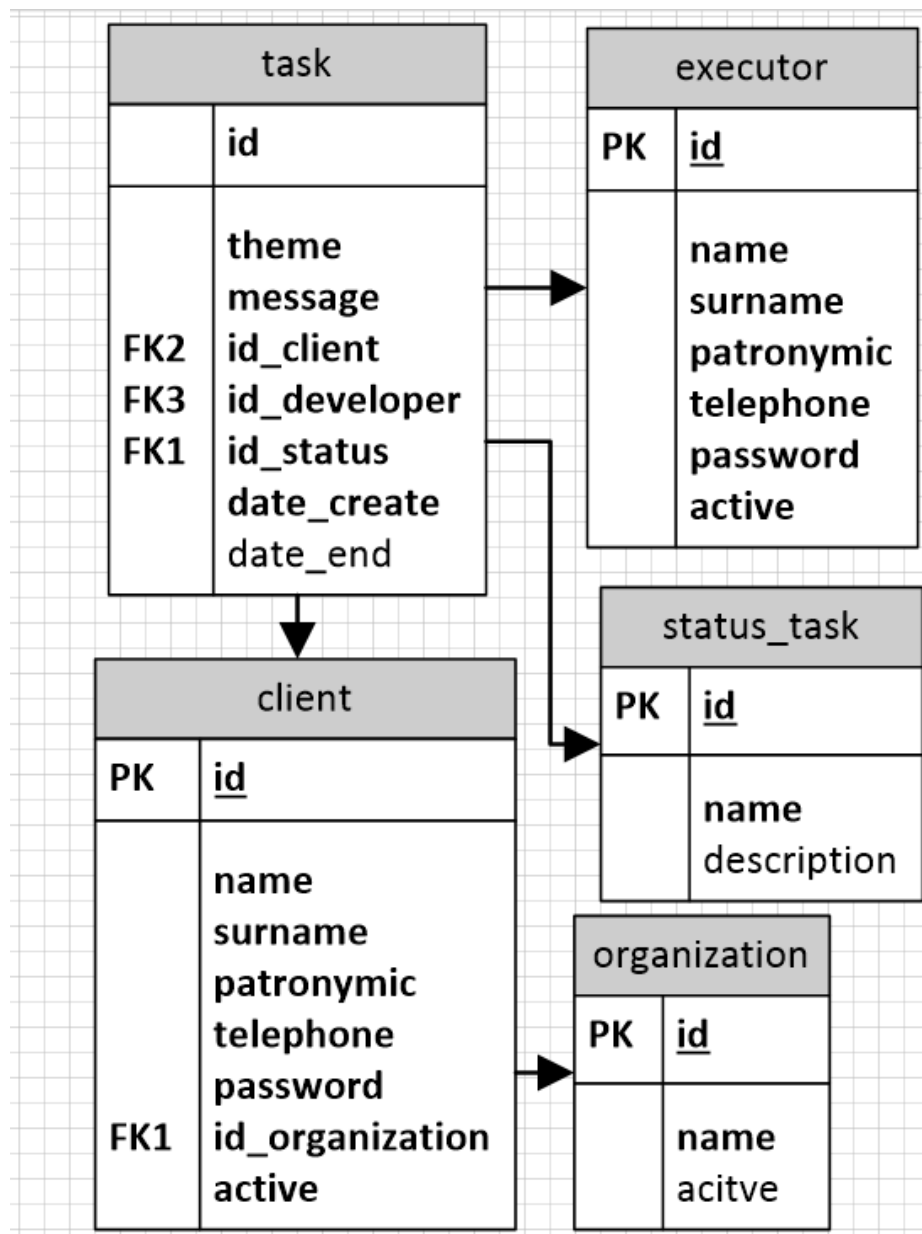


Рисунок 13 - Физическая модель ПМ

1.2 Обзор существующих методов и программных разработок в создании информационных систем медицинского назначения

Рассмотрим какие программные продукты существуют на рынке, которые могли бы оптимизировать процесс обработки обращения клиентов.

Для начала определяются критерии, по которым будет проводиться сравнение программных продуктов. Основные критерии включают в себя следующее:

- функциональность и возможности;
- удобство использования;
- простота в использовании;
- производительность;
- безопасность;
- стоимость;
- поддержка и обновления.

Ниже рассмотрим несколько вариантов программных продуктов.

YouTrack - коммерческая система отслеживания ошибок, программное обеспечение для управления проектами, разработанное компанией JetBrains.

YouTrack поддерживает поисковые запросы, автодополнение, манипуляцию с наборами задач, настройку набора атрибутов задачи, создание пользовательских рабочих процессов и реализует подход, основанный на преимущественном использовании клавиатуры.

YouTrack был разработан в соответствии с парадигмой языково-ориентированного программирования, использует JavaScript и Kotlin. Система использует встроенную базу данных H2 для записи и хранения данных. Для удалённых вызовов процедур использует REST-стиль [6].

Jira — коммерческая система отслеживания ошибок, предназначена для организации взаимодействия с пользователями, хотя в некоторых случаях используется и для управления проектами. Разработана компанией Atlassian, является одним из двух её основных продуктов. Имеет веб-интерфейс.

Jira имеет большое количество возможностей конфигурации: для каждого приложения может быть определён отдельный тип задачи с собственным workflow, набором статусов, одним или несколькими видами представления (англ. screens). Кроме того, с помощью так называемых «схем» можно опреде-

лить для каждого индивидуального Jira-проекта собственные права доступа, поведение и видимость полей и многое другое [6].

Trello — облачная программа для управления проектами небольших групп, разработанная Fog Creek Software.

Trello использует парадигму для управления проектами, известную как канбан, метод, который первоначально был популяризирован Toyota в 1980-х для управления цепочками поставок.

Trello ограничил поддержку тегов в виде десяти цветных меток, которые можно переименовать. Карточки поддерживают комментарии, вложения, сроки выполнения и контрольные списки. Trello имеет API. В настоящее время поддерживаются мобильные платформы приложений iPhone и Android. Также был разработан веб-сайт, чтобы быть доступным в большинстве мобильных веб-браузеров [6].

Ниже в таблице 1 представлено сравнение программных продуктов в области технической поддержки.

Таблица 1 - сравнение программных продуктов в области технической поддержки

Название	Удобство	Оплата	Поддержка	Безопасность	Простота
Trello	-	-	-	+	+
YouTrack	+	+	+	+	-
Jira	+	+	-	+	+

У каждого варианта есть свои недостатки и следующие ограничения:

- нет возможности изменять ПМ под свои нужды;
- существует зависимость от сторонних поставщиков;
- отсутствует полный контроль над данными и безопасностью, данные могут храниться или передаваться на сторонние сервисы;
- отсутствует возможности интеграции с другими системами.

Вследствие выше описанных проблем и с перспективой на будущее лучшим решением является разработка собственного программного продукта.

1.3. Планирование и определение затрат на разработку программного модуля ведения учетных записей медицинского центра

Для разработки ПМ требуется выполнить следующие этапы:

- анализ и проектирование ПМ;
- разработка базы данных ПМ;
- разработка серверной части ПМ;
- разработка интерфейсной части ПМ;
- тестирование разработанного ПМ.

На этапе анализа и проектирования ПМ выполняются следующие задачи:

- анализ предметной области;
- анализ задач, требуемых от ПМ;
- анализ средств разработки ПМ;
- проектирование логики работы ПМ;
- проектирование макета ПМ.

На этапе разработки базы данных ПМ выполняются следующие задачи:

- реализация сущностей базы данных;
- определение связей между сущностями;
- проставление индексов в базе данных.

На этапе разработки серверной части ПМ выполняются следующие задачи:

- настройка среды;
- разработка конечных точек;
- разработка классов серверов;
- разработка классов сущностей.

На этапе разработки интерфейсной части ПМ выполняются следующие задачи:

- разработка макета ПМ;
- разработка компонентов ПМ;
- разработка интерактивности в ПМ;

- разработка стилей в ПМ;
- разработка страниц в ПМ;

На этапе тестирования разработанного ПМ выполняются следующие задачи:

- тестирование скорости базы данных;
- тестирование контрольных точек серверной части ПМ;
- тестирование интерфейсной части ПМ.

Для выполнения описанных задач требуются следующие сотрудники:

- аналитик - специалист, занимающийся аналитическими исследованиями и обобщением в определенной сфере деятельности, который в совершенстве владеет методами анализа, обычно способен прогнозировать процессы и разрабатывать перспективные программы развития;

- проектировщик - специалист, занимающийся разработкой планов различных конструкций;

- разработчик базы данных - программировать, разрабатывать и внедрять системы баз данных;

- разработчик серверной части - это специалист, который занимается серверной частью сайтов. Он реализует внутреннюю логику работы приложения, обеспечивает его взаимодействие с базами данных и внешними сервисами.

- разработчик интерфейсов - это специалист, который занимается разработкой пользовательского интерфейса, то есть той части сайта или приложения, которую видят посетители страницы.

Перспектива разрабатываемого продукта заключается в следующих возможностях:

- возможность отправки сообщения на почту или номер телефона о выполненных работах;
- возможность уведомлять пользователей через браузер;
- возможность создавать для клиентов шаблоны текстов обращений;
- возможность создавать для исполнителей шаблоны текстов;

- повышение удобства программного продукта;
- функциональность изучения обращения клиентов на поиск похожих для достижения уменьшения нагрузки на тех. поддержку;
- добавление по необходимости новых маркеров обращения;
- функциональность определения исполнителя по сообщению и теме обращения;
- создание комментариев в обращении;
- возможность добавления файлов в обращении;
- функционал отчетов по проделанным работам.

Данные возможности со временем позволят увеличить прибыль за счет более качественной обратной связи с клиентами и улучшения качества программного продукта, что влечет за собой появление новых клиентов. Так же данные изменения облегчат работу отдела технической поддержки, что уменьшит количество необходимых сотрудников на данных должностях.

Управление проекта происходит с помощью программных средств.

На рисунке 14 представлен список задач анализа и проектирования.

<input type="checkbox"/> Анализ и проектирование		26 000,00р.
Анализ предметной области	Аналитик	10 000,00р.
Разработка ТЗ	Аналитик	10 000,00р.
Выбор средств разработки ПМ	Проектировщик	2 000,00р.
Проектирования макета ПМ	Проектировщик	4 000,00р.

Рисунок 14 - Список задач анализа и проектирования

На рисунке 15 представлен список задач разработки базы данных.

<input type="checkbox"/> Разработка базы данных		11 200,00р.
Разработка сущностей БД	Разработчик БД	5 600,00р.
Разработка связей	Разработчик БД	5 600,00р.

Рисунок 15 - Список задач разработки базы данных

На рисунке 16 представлен список задач разработки серверной части.

Разработка серверной части		52 000,00р.
Настройка среды	Разработчик серверной части	8 000,00р.
Разработка классов контроллеров	Разработчик серверной части	12 000,00р.
Разработка классов сущностей	Разработчик серверной части	16 000,00р.
Разработка классов серверов	Разработчик серверной части	16 000,00р.

Рисунок 16 - Список задач разработки серверной части

На рисунке 17 представлен список задач разработки клиентской части.

Разработка клиентской части		43 200,00р.
Разработка компонентов	Разработчик клиентской части	14 400,00р.
Разработка интерактивности	Разработчик клиентской части	14 400,00р.
Разработка страниц	Разработчик клиентской части	14 400,00р.

Рисунок 17 - Список задач разработки клиентской части

На рисунке 18 представлен список задач тестирования.

Тестирование		25 200,00р.
тестирования скорости базы данных	Тестировщик	8 400,00р.
тестирование серверной части	Тестировщик	8 400,00р.
тестирования интерфейсной части	Тестировщик	8 400,00р.

Рисунок 18 - Список задач тестирования

Лист ресурсов проекта представлен на рисунке 19.

Название ресурса	Тип	Стандартная ставка
Аналитик	Трудовой	250,00р./час
Проектировщик	Трудовой	250,00р./час
Разработчик БД	Трудовой	350,00р./час
Разработчик серверной части	Трудовой	500,00р./час
Разработчик клиентской части	Трудовой	450,00р./час
Тестировщик	Трудовой	350,00р./час

Рисунок 19 – Лист ресурсов проекта

На рисунке 20 представлен лист трудозатрат.

Название ресурса	Трудозатраты	Затраты
+ Не назначен	0 часов	0,00р.
+ Аналитик	80 часов	20 000,00р.
+ Проектировщик	24 часов	6 000,00р.
+ Разработчик БД	32 часов	11 200,00р.
+ Разработчик клиентской части	96 часов	43 200,00р.
+ Разработчик серверной части	104 часов	52 000,00р.
+ Тестировщик	72 часов	25 200,00р.

Рисунок 20 – Лист трудозатрат

Лист затрат на аналитика показан на рисунке 21.

- Аналитик	80 часов	20 000,00р.
Анализ предметной области	40 часов	10 000,00р.
Разработка ТЗ	40 часов	10 000,00р.

Рисунок 21 – Лист трудозатрат на аналитика

Лист затрат на проектировщика показан на рисунке 22.

- Проектировщик	24 часов	6 000,00р.
Выбор средств разработки ПМ	8 часов	2 000,00р.
Проектирования макета ПМ	16 часов	4 000,00р.

Рисунок 22 – Лист трудозатрат на проектировщика

Лист затрат на разработчика БД показан на рисунке 23.

- Разработчик БД	32 часов	11 200,00р.
Разработка сущностей БД	16 часов	5 600,00р.
Разработка связей	16 часов	5 600,00р.

Рисунок 23 – Лист трудозатрат на разработчика БД

Лист затрат на разработчика серверной части показан на рисунке 24.

▢ Разработчик серверной части	104 часов	52 000,00р.
Настройка среды	16 часов	8 000,00р.
Разработка классов	24 часов	12 000,00р.
Разработка классов	32 часов	16 000,00р.
Разработка классов серверов	32 часов	16 000,00р.

Рисунок 24 – Лист трудозатрат на разработчика серверной части

Лист затрат на разработчика клиентской части показан на рисунке 25.

▢ Разработчик клиентской части	96 часов	43 200,00р.
Разработка компонентов	32 часов	14 400,00р.
Разработка интерактивности	32 часов	14 400,00р.
Разработка страниц	32 часов	14 400,00р.

Рисунок 25 – Лист трудозатрат на разработчика клиентской части

Лист затрат на тестировщика показан на рисунке 26.

▢ Тестировщик	72 часов	25 200,00р.
тестирования скорости базы данных	24 часов	8 400,00р.
тестирование серверной части	24 часов	8 400,00р.
тестирования интерфейсной части	24 часов	8 400,00р.

Рисунок 26 – Лист трудозатрат на тестировщика

1.4 Техническое задание на разработку программного модуля ведения учетных записей медицинского центра

1.4.1 Общие сведения

1.4.1.1 Полное наименование системы и ее условное обозначение

Полное наименование программного модуля взаимодействия с клиентами
в медицинском центре.

Условное обозначение: ПМ ВКМЦ.

1.4.1.2 Плановые сроки начала и окончания работ

Начало работ: 14.04.2025г.

Окончание работ: 26.05.2025г.

1.4.2 Назначение и цели создания системы

1.4.2.1 Назначение системы

Система предназначена для автоматизации процесса работы технической
поддержки с клиентами в медицинском центре.

1.4.2.2 Цели создания системы

- повышение эффективности технической поддержки;
- улучшение коммуникации между клиентами и исполнителями;
- оптимизация процесса обработки обращений клиентов;
- обеспечение центрального хранилища обращений клиентов.

1.4.3 Характеристика объектов автоматизации

Объектами автоматизации являются процесс обработки обращений клиен-
тов, работа отдела технической поддержки.

1.4.4.1 Требования к системе в целом

1.4.4.1.1 Требования к структуре и функционированию системы

Система должна состоять из следующих подсистем:

- подсистема авторизации и аутентификации;
- подсистема управления обращениями;
- подсистема управления клиентами;
- подсистема управления исполнителями;

- подсистема управления организациями.

1.4.4.1.2 Требования к персоналу

Система предназначена для двух типов пользователей:

- клиент – сотрудники медицинского центра.
- исполнитель - сотрудники компании ООО «ИМЦ».

1.4.4.1.3 Требования к надежности

Требования к надежности программного обеспечения включают в себя следующие ключевые аспекты:

- безотказность – ПО должно выполнять свои функции без сбоев;
- отказоустойчивость – ПО должно сохранять возможность работоспособности при возникновении ошибок или сбоев;
- доступность - время, в течение которого система должна быть доступна для использования 24/7, не считая моментов технического обслуживания.

1.4.4.1.4 Требования к безопасности

- обеспечение конфиденциальности данных;
- предотвращение утечки данных;
- блокировка учетных записей при попытках несанкционированного доступа;
- наличие механизмов идентификации пользователей;
- наличие защиты от подбора паролей;
- наличие защиты от sql инъекций;
- наличие защиты от XSS уязвимостей.

1.4.4.1.5 Требования к эргономике и технической эстетике

ПМ должен иметь человеко-машинный интерфейс, удовлетворяющий следующим требованиям:

- взаимодействие системы и пользователя должно осуществляться на русском языке, за исключением системных сообщений, не подлежащих русификации;

- при работе с интерфейсом пользователь должен быть ориентирован на работу с клавиатурой и манипулятором графической информации «мышь»;
- должно быть реализовано отображение на экране только тех возможностей, которые доступны конкретному пользователю в соответствии с его функциональной ролью в системе;
- должна быть реализована возможность работы с системой при двух мониторной конфигурации дисплеев;
- представление управляющих элементов, экранных форм и их информационных элементов (окон, панелей и т.п.) должно быть унифицировано. Экранные формы должны полностью находиться в видимой площади экрана монитора с диагональю 17' при разрешении экрана 1280 x 1024 и выше.

1.4.4.2 Требования к функциям, выполняемым системой

1.4.4.2.1 Подсистема авторизации и аутентификации

- авторизация клиента и исполнителя;
- аутентификация клиента и исполнителя.

1.4.4.2.2 Подсистема управления обращениями

- создание обращения клиентом;
- изменение обращения;
- изменение статуса обращения;
- просмотр всех обращений для исполнителей;
- фильтрация обращений.

1.4.4.2.3 Подсистема управления пользователями

- добавление клиента;
- изменение активности клиента;
- получение списка клиентов;
- добавление исполнителя;
- изменение активности исполнителя;
- получение списка исполнителей.

1.4.4.2.4 Подсистема управления организациями

- добавление организации клиента;
- получение списка организаций клиентов;
- изменение активности организаций.

1.4.4.3 Требования к видам обеспечения

1.4.4.3.1 Требования к информационному обеспечению

Информационное обеспечение должно обеспечивать системность, информационную полноту, избирательность, непрерывность, целостность потока информации по всей совокупности релевантной информации об объектах контроля и субъектах наблюдения.

Уровень хранения данных в ПМ должен быть построен на основе реляционных СУБД.

Для обеспечения целостности данных должны использоваться встроенные механизмы СУБД.

Технические средства, обеспечивающие хранение информации, должны использовать современные технологии, позволяющие обеспечить повышенную надежность хранения данных и оперативную замену оборудования.

1.4.4.3.2 Требования к программному обеспечению

На сервере должны быть установлены следующие ПО:

- node v 20.12.2 – для развертывания клиентской и серверной части программного модуля;
- npm v10.5.0 - менеджер пакетов;
- python – язык программирования, который используется для сборки программ в node;
- vue-cli-service – библиотека node, которая используется для сборки клиентской части программного модуля;
- nest - фреймворк node, который используется для сборки серверной части программного модуля;
- PostgreSQL – база данных для хранения данных.

У клиента должен быть установлен один из ниже описанных браузеров последних версий: Яндекс, Chromium-Gost.

1.4.4.3 Требования к техническому обеспечению

Сервер должен соответствовать рекомендуемым требованиям программно-го обеспечения и поддерживать технологию RAID1.

Персональные компьютеры клиентов должны иметь следующие минимальные характеристики:

- ОП 8Гб;
- процессор 4 ядерный с 3ГГц.

1.4.5 Состав и содержание работ по созданию программного модуля

- анализ и проектирование ПМ;
- разработка базы данных ПМ;
- разработка серверной части ПМ;
- разработка интерфейсной части ПМ;
- тестирование разработанного ПМ.

1.4.6 Порядок контроля и приемки системы

Тестирование ПМ будет осуществлять поочередно:

- тестирование серверной части ПМ;
- тестирование компонентов интерфейса ПМ;
- целостное тестирование ПМ.

Ответственность за организацию и проведение приемки системы должен нести заказчик. Приемка ПМ должна производиться по завершению всех задач ПМ. При этом необходимо предоставить обеспечение материальной частью, проектной документацией и специально выделенным персоналом.

На завершающем этапе при приемке системы должно быть составление акта приемки.

1.4.7 Требования к документированию

Требуется разработать следующие документы:

- руководство клиентам;
- руководство исполнителей;

Документация должна быть выполнена на русском языке.

Электронная версия документации предоставляется в форматах PDF и DOC/DOCX.

Требования к содержанию руководства пользователя:

- описание интерфейса ПМ;
- описание выполняемых действий пользователями в ПМ.

1.4.8 Источники разработки

- ГОСТ 34.602-20 "Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы";
- <https://nodejs.org> – документация node;
- <https://nestjs.com> - документация фреймворка серверной части программного модуля;
- <https://vuejs.org> - документация фреймворка клиентской части программного модуля;
- <https://www.postgresql.org> - документация базы данных программного модуля.

В рамках данной главы проведен анализ предприятия ООО «ИМЦ», определена организационная структура предприятия ООО «ИМЦ», разработаны диаграммы IDFE0 процесса обработки обращения клиентов, разработана модель ТО-ВЕ процесса обработки обращения клиентов с использованием ПМ, разработаны диаграммы DFD и IDEF3, разработана физическая модель ПМ, произведен анализ аналогов требуемого ПМ, произведен выбор разработки собственного ПО, определены этапы и задачи процесса разработки ПМ, определены ресурсы процесса разработки ПМ, определены затраты на разработку ПМ, разработано ТЗ для разработки ПМ.

2 ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО МОДУЛЯ АДМИНИСТРИРОВАНИЯ

2.1 Проектирование программного модуля ведения учетных записей медицинского центра с использованием объектного моделирования

Программный модуль будет разработан для информационно-технической деятельности отдела технической поддержки ООО «ИМЦ» для оптимизации процесса обработки обращения клиента.

Задача программного модуля - систематизировать поступающие обращения клиентов и оптимизировать процесс их обработки.

Входными данными программного модуля являются данные клиента и его обращение.

Выходными данными программного модуля является выполненное обращение клиента.

Сущности программного модуля:

- клиент;
- исполнитель;
- обращение (задача);
- организация клиента.

Программный модуль выполняет следующие функции:

- оформление обращений клиентов;
- добавление клиента;
- получение списка клиентов;
- получение информации об клиенте;
- добавление исполнителя;
- редактирование исполнителя;
- получение списка исполнителей;
- добавление организации клиента;

- получение списка организаций клиентов;
- редактирование организации;
- возможность авторизации клиента и исполнителя;
- возможность аутентификации клиента и исполнителя;
- возможность просмотра всех обращений;
- создание обращения;
- изменение статуса обращения;
- фильтрация по обращениям;
- изменение обращения;
- валидация изменения статусов обращения.

Принципиальное различие между структурным и объектно-ориентированным подходом заключается в способе декомпозиции системы. ОО подход использует объектную декомпозицию, при этом статическая структура системы описывается в терминах объектов и связей между ними, а поведение системы описывается в терминах обмена сообщений между объектами [8].

UML предоставляет средства для создания визуальных моделей, которые единообразно понимаются всеми разработчиками, вовлеченными в проект, и являются средством коммуникации в рамках проекта. Диаграмма в UML - это графическое представление набора элементов. Диаграммы рисуют для визуализации системы с разных точек зрения. При визуальном моделировании на UML используются восемь видов диаграмм, каждая из которых может содержать элементы определенного типа [7].

Выбор объектно-ориентированного подхода вместо структурного обоснован следующими причинами:

- объектно-ориентированные системы лучше моделируют предметную область. Они проще адаптируются к изменяющимся условиям, легче изменяются, устойчивее и позволяют создавать более крупные проекты;

– объектная декомпозиция уменьшает размер программных систем. Это достигается за счёт повторного использования общих механизмов, что приводит к существенной экономии выразительных средств;

– объектно-ориентированные системы снижают риск при создании сложной программной системы. Она развивается из меньших систем, в которых уже уверены;

– использование объектного подхода повышает уровень унификации разработки и пригодность для повторного использования.

Диаграмма вариантов использования — это диаграмма, на которой изображаются отношения между актерами и вариантами использования.

Ниже на рисунке 27 представлена диаграмма вариантов использования ПМ.

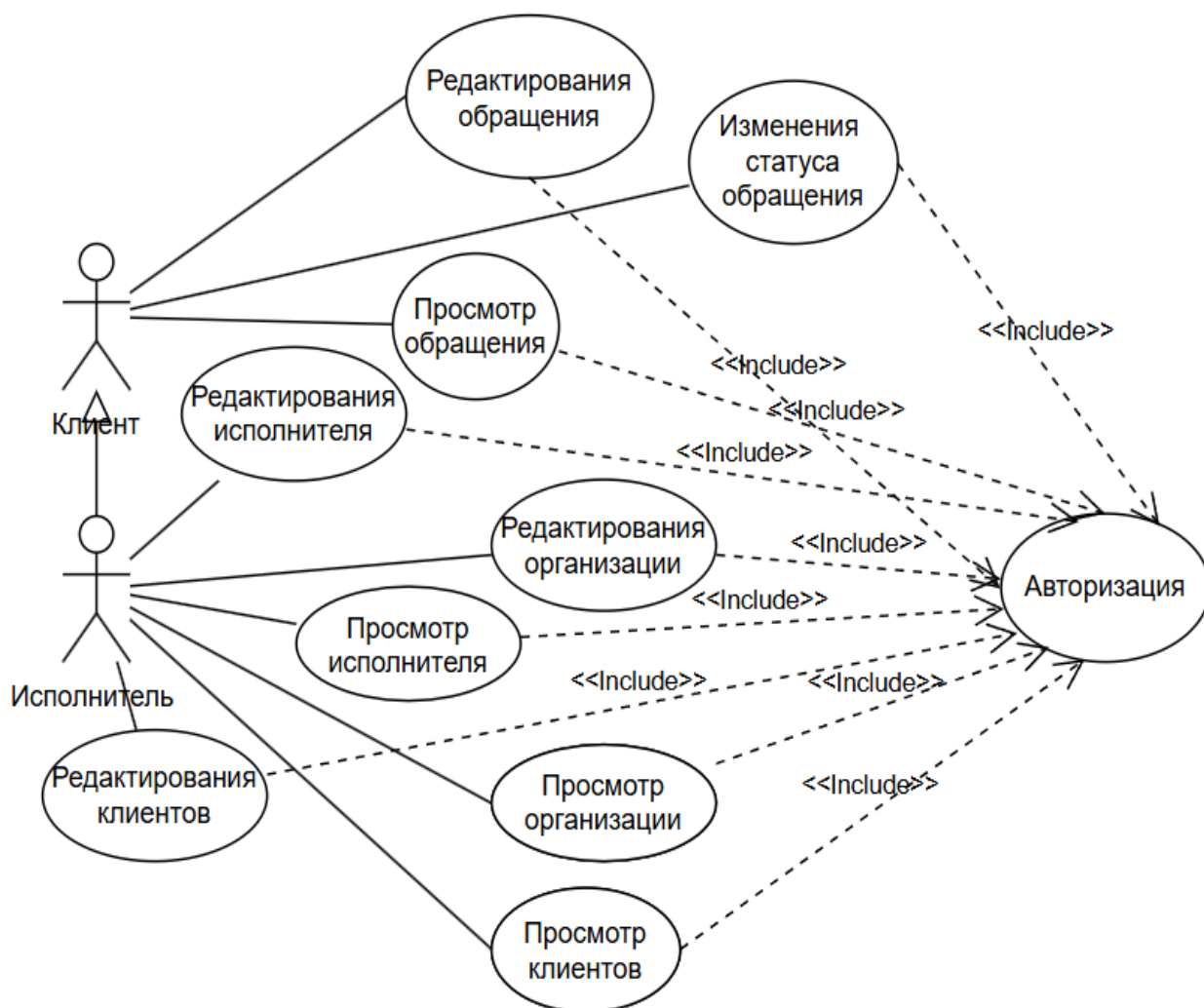


Рисунок 27 - Диаграмма вариантов использования ПМ

Диаграмма классов — это структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей (отношений) между ними.

Ниже на рисунке 28 представлена диаграмма классов ПМ.

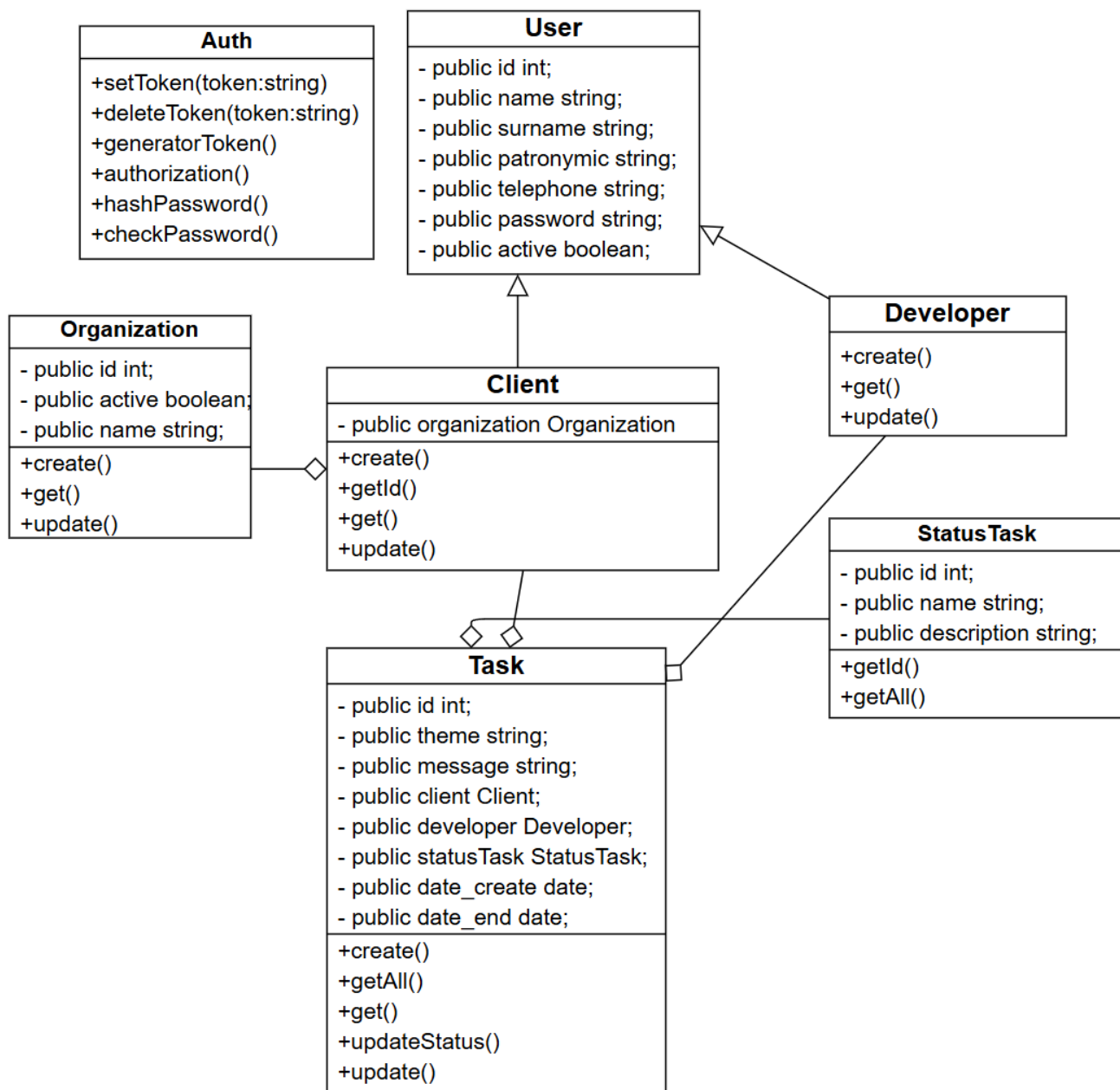


Рисунок 28 - Диаграмма классов ПМ

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

На рисунке 29 представлена диаграмма последовательности ПМ.

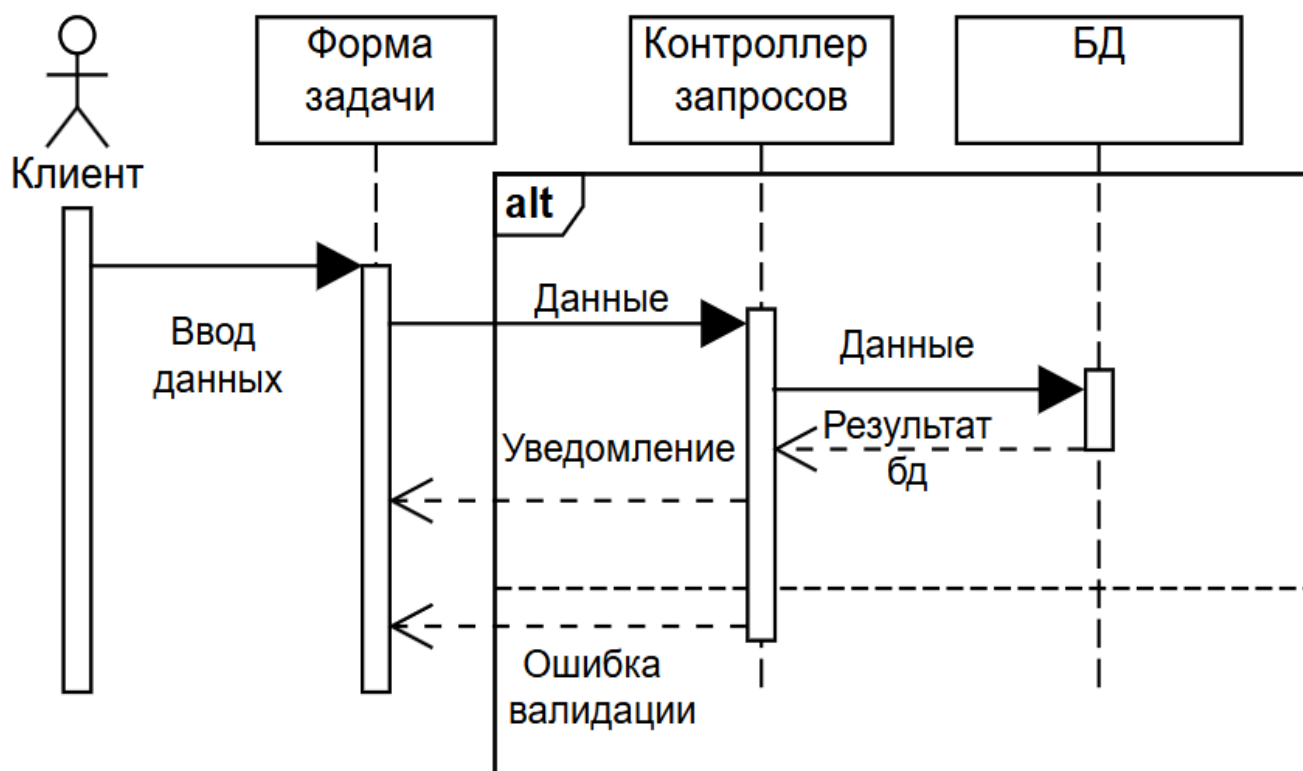


Рисунок 29 - Диаграмма последовательности ПМ

Диаграмма компонентов - это структурная диаграмма языка унифицированного моделирования, она описывает особенности физического представления системы. Диаграмма компонентов позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами.

На рисунке 30 представлена диаграмма компонентов ПМ.

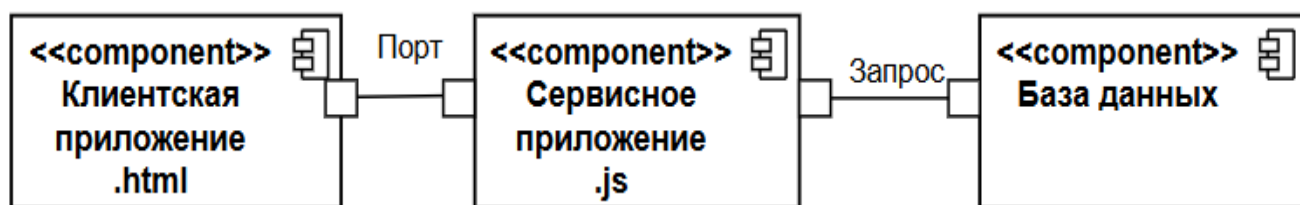


Рисунок 30 - Диаграмма компонентов ПМ

Диаграмма развертывания предназначена для визуализации элементов и компонентов программы, существующих лишь на этапе ее исполнения. При этом представляются только компоненты-экземпляры программы, являющиеся исполнимыми файлами или динамическими библиотеками.

На рисунке 31 представлена диаграмма развертывания ПМ.

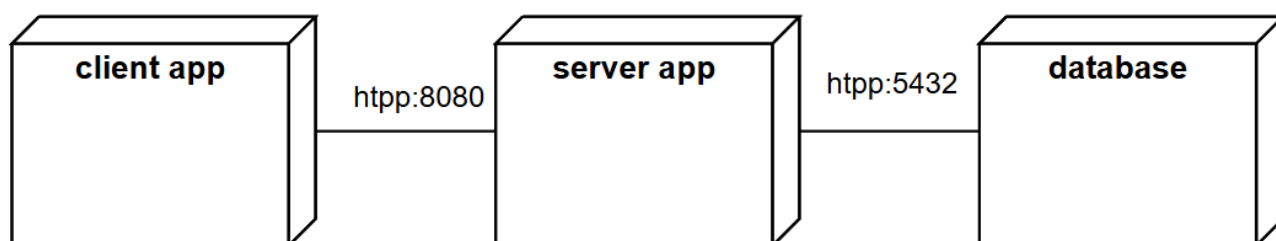


Рисунок 31 - Диаграмма развертывания ПМ

Диаграмма состояний — ориентированный граф для конечного автомата, в котором вершины обозначают состояния дуги показывают переходы между двумя состояниями. На практике вершины обычно изображаются в виде окружностей и, если нужно, двойных окружностей.

На рисунке 32 представлена диаграмма состояния ПМ.

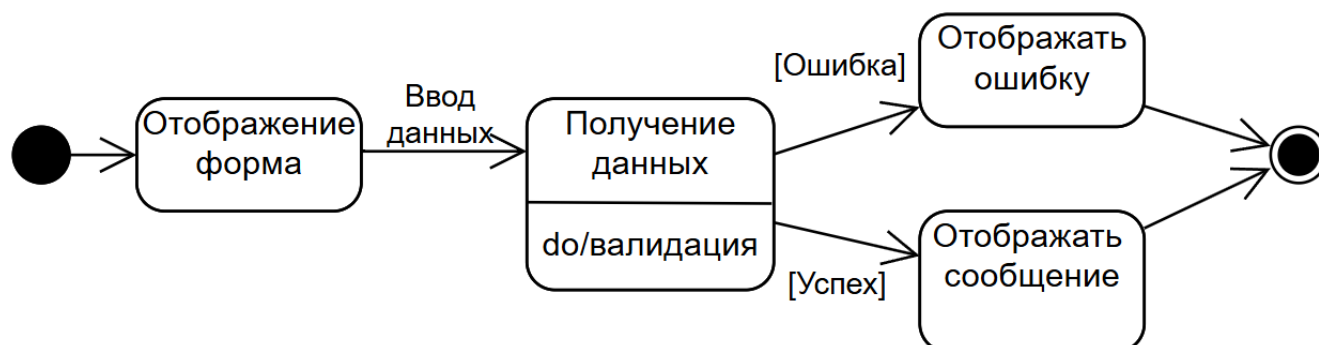


Рисунок 32 - Диаграмма состояния ПМ

2.2 Выбор языка и среды разработки программного модуля ведения учетных записей медицинского центра

На начальных этапах процесса проектирования программного модуля необходимо принять принципиальные решения, во многом определяющие этот процесс, а также качество и трудоемкость разработки. К таким решениям относятся:

- выбор архитектуры программного обеспечения;
- выбор типа пользовательского интерфейса;
- выбор подхода к разработке;
- выбор языка и среды программирования.

Языки веб-разработки - это языки программирования и технологии, используемые для создания веб-сайтов, веб-приложений и компонентов серверной части. Они составляют основу веб-разработки и определяют характер взаимодействия пользователей с веб-контентом [8]. Существует два основных аспекта веб-разработки:

Front-End Development: Включает в себя проектирование и реализацию визуальных элементов, которые видят пользователи и с которыми они взаимодействуют в веб-приложении. Разработчики фронт-энда используют комбинацию HTML, CSS и JavaScript для создания визуально привлекательного и функционального пользовательского интерфейса.

Back-End Development: Работает с компонентами на стороне сервера и манипулирует данными, обеспечивая их хранение, обработку и получение. Это обеспечивает бесперебойную работу веб-приложения и его масштабирование при необходимости. К распространенным языкам back-end относятся Python, PHP, Ruby, Java и C#.

Для Front-End Development в качестве языка будет использоваться JavaScript как один из самых лидирующих языков в Front-End разработке.

Кроме языка программирования требуется выбрать используемые фреймворк.

Фреймворк (англ. framework — «каркас, структура») — это готовый набор инструментов, который помогает разработчику быстро создать продукт: сайт, приложение, интернет-магазин, CMS-систему.

В таблице 2 представлено сравнение Фреймворк для Front-End разработки.

Таблица 2 – Сравнение Фреймворк для Front-End разработки

Фреймворк	Производительность	Сложность	Комьюнити
Angular	-	Сложный	+
Vue3	+	Простой	-
React	+	Средней	+
Svelte	-	Простой	-

В качестве выбора предпочтителен Vue3.

Vue3 сочетает в себе простоту изучения и использования, за счет своего меньшего размера и улучшения производительности в 3 версии выигрывает у своих аналогов, также имеет большое и активное сообщество разработчиков и широкий выбор инструментов и библиотек.

Существует множество способов писать код для веб-приложений: от текстовых редакторов до облачных сред разработки.

IDE — это набор программных инструментов, которые используются для создания ПО. [11]

Редактор кода - это программное обеспечение, которое позволяет программистам и разработчикам создавать, редактировать и отлаживать исходный код программ. [11]

В таблице 3 представлено сравнение средств для создания веб-приложения.

Таблица 3 – Сравнение средств для создания веб-приложения

Название	Платность	Удобность	Быстрота
VS code	-	+	+
WebStorm	+	+	-
Sublime Text	-	-	+

Из выбранных вариантов предпочтителен VS code.

Visual Studio Code сочетает в себе простоту редактора исходного кода с мощными инструментами разработчика, такими как доработка и отладка кода IntelliSense.

В основе Visual Studio Code лежит молниеносный редактор исходного кода, идеально подходящий для повседневного использования. Благодаря поддержке сотен языков VS Code позволяет мгновенно повысить производительность при работе с подсветкой синтаксиса, подбором скобок, автоматическим отступом, выделением блоков, фрагментами и многим другим.

Система управления базами данных (СУБД) — это программное обеспечение, предназначенное для создания, управления, обновления и анализа баз дан-

ных. Она обеспечивает интерфейс для взаимодействия пользователя или приложения с данными, хранящимися в базе данных. СУБД позволяют структурировать данные таким образом, чтобы обеспечить их легкий доступ, безопасность и эффективное использование.

В таблице 4 представлено сравнение СУБД.

Таблица 4 – Сравнение СУБД

СУБД	Бесплатность	Размер базы	Размер таблицы	Число пользователей
HSQldb	нет	28 TB	120 GB	Не ограничено
Microsoft SQL Server	нет	16 TB	532 GB	Не ограничено
MySQL	да	256 TB	256 TB	Не ограничено
PostgreSQL	да	Неограничен	32 TB	Не ограничено

Лучшим вариантом СУБД является PostgreSQL, который с увеличением количества данных не будет иметь ограничений на размеры БД, будет относиться к бесплатной СУБД и предоставлять возможность быстрого чтения среди множества данных и неограниченный размер хранения индексов.

PostgreSQL заработал прочную репутацию благодаря своей проверенной архитектуре, надежности, целостности данных, надежному набору функций, расширяемости и преданности сообщества открытого исходного кода, стоящего за программным обеспечением, для последовательного предоставления производительных и инновационных решений.

PostgreSQL работает во всех основных операционных системах, совместим с ACID с 2001 года и имеет мощные надстройки, такие как популярный расширитель геопространственных баз данных PostGIS. Неудивительно, что PostgreSQL стала предпочитаемой реляционной базой данных с открытым исходным кодом для многих людей и организаций. [12]

Backend (бэкенд) – серверная часть сайта. Отвечает за быструю загрузку страниц, обработку данных, безопасность, интеграцию с другими системами.

В таблице 5 представлено сравнение фреймворков для разработки веб сервиса.

Таблица 5 – Сравнение фреймворков для разработки веб сервиса

Фреймворк	Сложность	Документация	Производительность
Nest(js)	Средний	+	+
Django(python)	Легкий	-	-
Spring(java)	Сложный	-	+

Из представленных Фреймворков выбран Nest(js) из-за своей простоты и хорошо написанной документации в сравнение с др. фреймворками, так же используется тот же язык что и в frond-end разработке, что значительно упрощает разработку.

В рамках данной главы были определены сущности ПМ, определены задачи выполняющие ПМ, разработаны диаграммы вариантов использования ПМ, классов ПМ, последовательности ПМ, компонентов ПМ, состояния ПМ, выбраны языки и фреймворки разработки front-end и back-end приложения, а также система управления базой данных и программные средства для разработки веб приложения.

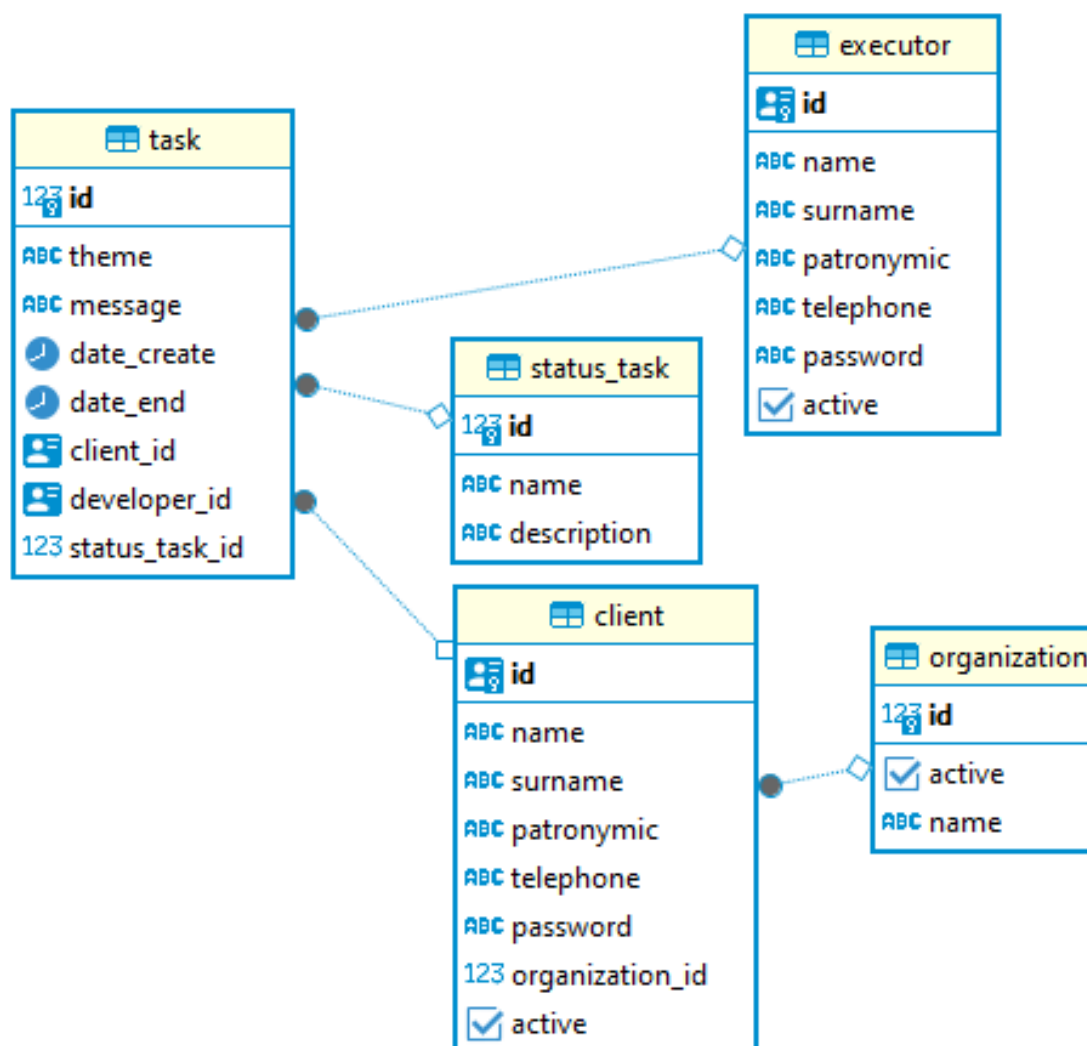
3. РЕАЛИЗАЦИЯ ПРОГРАММНОГО МОДУЛЯ ВЕДЕНИЯ УЧЕТНЫХ ЗАПИСЕЙ МЕДИЦИНСКОГО ЦЕНТРА.

3.1. Реализация программного модуля ведения учетных записей медицинского центра.

Реализация ПМ предусматривает создание программного продукта, состоящего из нескольких взаимосвязанных компонентов. Архитектура системы построена по принципу разделения на серверную часть и пользовательский интерфейс. Такой подход обеспечивает гибкость развития и поддержки приложения, а также позволяет масштабировать систему при необходимости.

Для реализации серверной части необходимо разработать базу данных.

На рисунке 33 представлена схема базы данных ПМ.



Рисунке 33 - Схема базы данных ПМ

Далее необходимо описать каждую сущность. Это позволит понять структуру и логику хранения данных, и определить назначения каждого атрибута.

На таблице 6 представлена сущность client.

Таблица 6 - Сущность client

Поле	Тип данных	Примечание
id	int	Первичный ключ
name	varchar	Имя пользователя
surname	varchar	Фамилия пользователя
patronymic	varchar	Отчество пользователя
id_contract	int	Внешний ключ
telephone	varchar	Номер телефона

Сущность организация представлена в таблице 7.

Таблица 7 - Сущность организация

Поле	Тип данных	Примечание
id	int	Первичный ключ
name	varchar	Имя организации
active	boolean	Активность поддержки организации

Сущность разработчик представлена в таблице 8.

Таблица 8 - Сущность исполнитель

Поле	Тип данных	Примечание
id	int	Первичный ключ
name	varchar	Имя разработчика
surname	varchar	Фамилия разработчика

Сущность статус задачи представлена в таблице 9.

Таблица 9 - Сущность статус задачи

Поле	Тип данных	Примечание
id	int	Первичный ключ
name	varchar	Название статуса задачи
description	text	Описание статуса задачи

Сущность задача представлена в таблице 10.

Таблица 10 - Сущность задача

Поле	Тип данных	Примечание
id	int	Первичный ключ
theme	varchar	Тема задачи
message	text	Сообщение задачи
id_client	int	Внешний ключ
id_developer	int	Внешний ключ
id_status	int	Внешний ключ
date_create	timestamp	Дата и время создание задачи

После разработки базы данных необходимо разработать серверную часть приложения, которое будет выполнять логику и взаимодействовать с базой данных и возвращать данные клиентской частью приложения для вывода информации.

Серверная часть представлена в виде REST API.

Подход REST API представляет собой набор конечных точек, которые выполняют определенный процесс ПМ.

Ниже представлен код главного файла серверной части.

```
import { NestFactory } from '@nestjs/core';
import { AppModule } from './app.module';
import { SwaggerModule } from '@nestjs/swagger';
import { SwaggerConfig } from './lib/config/swagger.config';
import { ValidationPipe } from '@nestjs/common';
async function bootstrap() {
  const app = await NestFactory.create(AppModule);
  app.enableCors();
  app.useGlobalPipes(new ValidationPipe({ transform: true }));
  const document = SwaggerModule.createDocument(app, SwaggerConfig);
  SwaggerModule.setup('api', app, document);
}
```

```

    await app.listen(3000); }

bootstrap();

Ниже представлен код класса контроллер задач

import { BadRequestException, Body, Controller, Get, Post, Put, Query, Req,
UseGuards } from '@nestjs/common';

import { ApiBearerAuth, ApiQuery, ApiTags } from '@nestjs/swagger';
import { TaskCreateDTO } from './dto/task.create.dto';
import { TaskService } from './task.service';
import { TypeUserDecorator, UserDecorator } from
'src/lib/decorator/user.decorator';

import { AuthGuard } from 'src/auth/auth.guard';
import { ClientEntity } from 'src/client/client.entity';
import { TaskUpdateDTO } from './dto/task.update.dto'; @Controller("tasks")
@ApiTags("task")
@UseGuards(AuthGuard)
@ApiBearerAuth()
export class TaskController {

    constructor(private readonly taskService: TaskService) { }

    @UserDecorator(TypeUserDecorator.client)
    @Get("/id")
    @ApiQuery({ name: 'id', required: true })
    public async getId(@Query("id") id?: string) {
        const resBd = await this.taskService.getId(+id);
        return await this.taskService.convertTask([resBd])[0];
    }

    @UserDecorator(TypeUserDecorator.client)
    @Put("/status")
    @ApiQuery({ name: 'status_id', required: true })
    @ApiQuery({ name: 'task_id', required: true })

```



```

    public async updateStatus(@Req() req: any, @Query("status_id") statusId?:
string, @Query("task_id") taskId?: string ) {
        const user: ClientEntity = req?.user.user;
        return await this.taskService.updateStatus(+statusId, +taskId, user.id);
    }
    @UserDecorator(TypeUserDecorator.client)
    @Put()
    public async update(
        @Body() taskUpdateDTO: TaskUpdateDTO
    ) { return await this.taskService.update(taskUpdateDTO); }
}

```

Ниже представлен код класса сервис задач

```

import { BadRequestException, Injectable } from '@nestjs/common';
import { InjectRepository } from '@nestjs/typeorm';
import { TaskEntity } from './task.entity';
import { Repository } from 'typeorm';
import { TaskCreateDTO } from './dto/task.create.dto';
import { TaskUpdateDTO } from './dto/task.update.dto';
@Injectable()
export class TaskService {
    constructor(
        @InjectRepository(TaskEntity)
        private taskRepository: Repository<TaskEntity>,
    ) { }
    public convertTask(taskEntity: TaskEntity[]) {
        const result = [];
        for (const e of taskEntity) {
            result.push({ id: e.id, theme: e.theme, message: e.message, date_create:
e.date_create, date_end: e.date_end, client_name: e.client.name, developer_name:
e.executor.name, status_name: e.statusTask.name, executor_id: e.executor.id })

```

```

    }
    return result;
}

public async create(taskCreateDTO: TaskCreateDTO, userId: string) {
    const taskEntity = { statusTask: { id: 1 }, client: { id: userId }, executor: {
id: taskCreateDTO.executor_id }, message: taskCreateDTO.message, theme: taskCre-
ateDTO.theme };

    return await this.taskRepository.save(taskEntity);
}

public async get(statusId: number, clientId: string, executorId: string) {
    const resBd = await this.taskRepository.find({
        relations: ["client", "executor", "statusTask"],
        where: this.genetatorWhereGet(statusId, clientId, executorId)
    });
    return this.convertTask(resBd);
}
}

```

Ниже представлен код класса сущность задач

```

import { ClientEntity } from "src/client/client.entity";
import { ExecutorEntity } from "src/executor/executor.entity";
import { StatusTaskEntity } from "src/statusTask/statusTask.entity";
import { Column, Entity, JoinColumn,ManyToOne, PrimaryGeneratedColumn
} from "typeorm";

@Entity("task")
export class TaskEntity {
    @PrimaryGeneratedColumn()
    id: number;

    @Column()
    theme: string;

    @Column() message: string;
}

```

```

@ManyToOne(() => ClientEntity, (client) => client.id)
@JoinColumn({ name: "client_id" }) client: ClientEntity
@ManyToOne(() => ExecutorEntity, (executor) => executor.id)
@JoinColumn({ name: "executor_id" }) executor: ExecutorEntity
@ManyToOne(() => StatusTaskEntity, (statusTask) => statusTask.id)
@JoinColumn({ name: "status_task_id" })

statusTask: StatusTaskEntity

@Column({ type: 'date', default: () => 'CURRENT_DATE' })
date_create: Date;

@Column({ type: 'date', nullable: true }) date_end: Date;
}

```

Подобные классы реализованы аналогичным принципом.

На рисунке 34 представлены конечные точки группы auth.

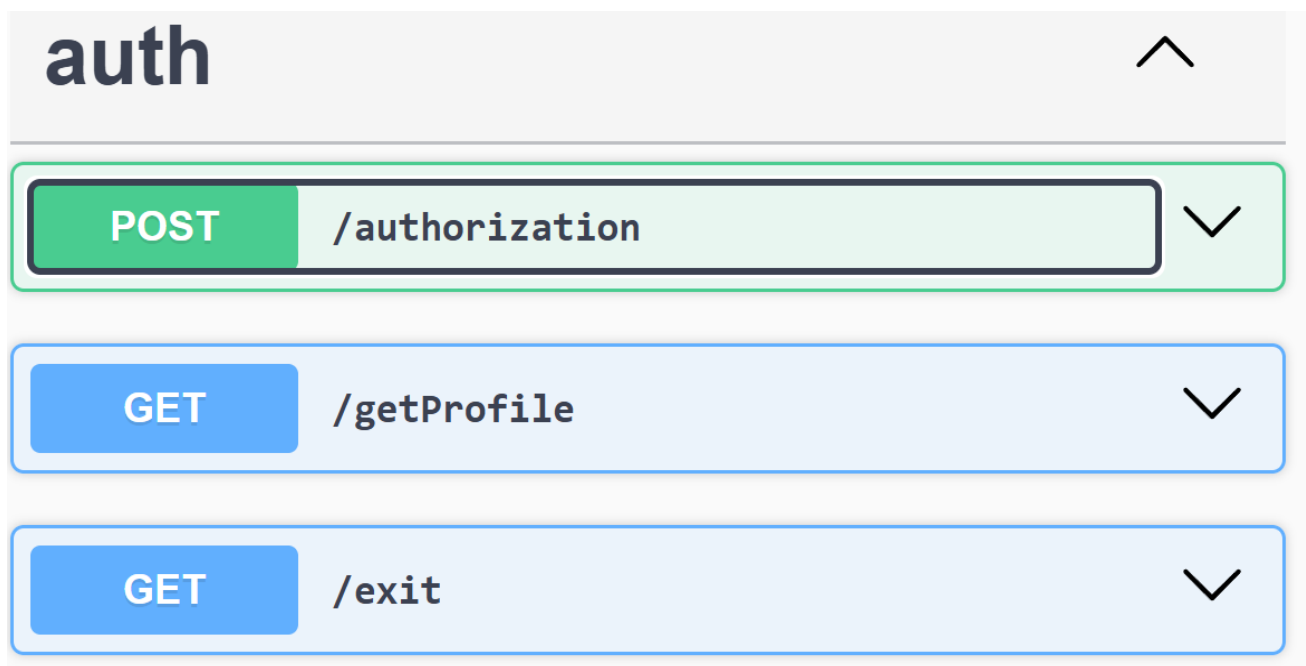


Рисунок 34 – Конечные точки группы auth

authorization POST - конечная точка для авторизации клиентов и исполнителей путем ввода логина и пароля.

getProfile GET - конечная точка для получение личных данных авторизованных клиентов и исполнителей.

exit GET - конечная точка для выхода из системы.

На рисунке 35 представлены конечные точки группы task.

task

POST

/taks

🔒

▼

GET

/taks

🔒

▼

PUT

/taks

🔒

▼

GET

/taks/all

📋

🔒

▼

GET

/taks/id

🔒

▼

PUT

/taks/status

🔒

▼

Рисунок 35 - Конечные точки группы task

task POST - конечная точка для создания обращения клиентов.

task GET - конечная точка для получения задач клиентов, клиентами ограничивает видимость задач по идентификатору клиента.

Task PUT - конечная точка для изменения задачи.

task/all GET – конечная точка для получения задач клиентов, исполнителями, позволяет видеть все задачи.

task/id GET - конечная точка для получения конкретной 1 задачи по идентификатору.



task/status PUT – конечная точка для изменения статуса задачи.

На рисунке 36 представлены конечные точки группы executor.

executor



POST

/executor



PUT

/executor



GET

/executor





Рисунок 36 - Конечные точки группы executor

executor POST – конечная точка для добавления нового исполнителя.
executor GET – конечная точка для получения списка исполнителя.
executor PUT – конечная точка для изменения активность исполнителя.

На рисунке 37 представлены конечные точки группы StatusTask.

StatusTask

GET

/StatusTask




Рисунок 37 - Конечные точки группы StatusTask

StatusTask GET - конечная точка для получения списка статусов задач.

На рисунке 38 представлены конечные точки группы client.


client			
POST	/client		 
GET	/client		
PUT	/client		
GET	/client/id		

Рисунок 38 - Конечные точки группы client

client POST конечная точка для добавления нового клиента доступно только авторизованным исполнителям.

client GET - конечная точка для получения списка клиентов доступно только авторизованным исполнителям.

client/id GET - конечная точка для получения клиента по id доступно только авторизованным исполнителям.

На рисунке 39 представлены конечные точки группы organization.







organization			
POST	/organization		
PUT	/organization		
GET	/organization		

Рисунок 39 - Конечные точки группы organization

Organization POST - конечная точка для создания организации.

Organization PUT - конечная точка для изменения активности организации.

Organization GET - конечная точка для получения списка организации.

3.2. Описание интерфейса программного модуля ведения учетных записей медицинского центра

После разработки серверной части ПМ необходимо разработать интерфейс ПМ, расположение элементов в и его функциональность.

ПМ является закрытым ПО с ограниченным доступом, поэтому для получения доступа к ПМ необходимо произвести процесс авторизации пользователя путем ввода номера телефона и пароля. Для этого требуется разработать экран авторизации, представленный на рисунке 40.

Форма авторизации

Номер телефона

Пароль

Исполнитель ☐

Рисунок 40 - Экран авторизации

В данном экране пользователь должен ввести номера телефона и пароля от своего аккаунта и нажать на кнопку войти, после этого при успешном вводе серверная часть вернет токен авторизации для работы в ПМ, при неудачном вводе появится сообщение об ошибке.

На рисунке 41 представлен компонент навигационное меню для исполнителей.

Клиенты Исполнители **Задачи** Организация Выход

Рисунок 41 - Компонент навигационное меню для исполнителей

На рисунке 42 представлен компонент навигационное меню для клиентов.

Рисунок 42 - Компонент навигационное меню для клиентов

Данный компонент позволяет обеспечивать удобные переходы между разными разделами ПМ.

Далее представлен основной экран ПМ задачи.

Данный экран представляет собой возможность просматривать задачи в таблице, фильтрация задач для более удобного поиска и возможность создавать новую задачу.

На рисунке 43 представлен экран задачи.

Фильтр задач:

Статус задачи

Исполнитель

Поиск

ID	Тема	Дата задачи	Дата завершения	Клиент	Исполнитель	Статус	
37	Задача тест2	2025-04-26		2		Отмененна	Открыть заявку
36	Задача тест1	2025-04-26		2		Создана	Открыть заявку

Создание задачи:

Тема задачи

Сообщение задачи

Исполнитель

Создать задачу

Рисунок 43 - Экран задачи

На данном экране представлен компонент фильтр задач с кнопкой поиска который обновить таблицу.

В таблице есть кнопка открыть заявку позволяет более детально рассмотреть задачу и изменить статус задачи с помощью модального окна с задачей.

На рисунке 44 представлено модальное окно с задачей.

Задача: #36

Тема задачи

Сообщение задачи

Исполнитель

Рисунок 44 - Модальное окно с задачей

В данном модальном окне есть возможность изменить содержимое задачи и представлены наборы кнопок с помощью которых происходит изменения статуса задач.

На рисунке 45 представлен экран исполнители для исполнителей.

ID	Имя	Фамилия	Отчество	Телефон	Активность	
71129101-fde2-436f-a60b-fb267bdd521f	admin	admin	admin	1	true	<input type="button" value="Изменить активность"/>

Создание разработчика:

Имя разработчика

Фамилия разработчика

Отчество разработчика

телефон разработчика

пароль разработчика

Рисунок 45 - Экран исполнители для исполнителей

Экран представляет возможность просмотра исполнителей в таблице, изменения активности исполнителей и создания новых исполнителей.

На рисунке 46 представлен экран исполнители для клиентов.

ID	Имя	Фамилия	Отчество	Телефон	Активность
71129101-fde2-436f-a60b-fb267bdd521f	admin	admin	admin	1	true

Рисунок 46 - Экран исполнители для клиентов

Экран исполнители для клиентов позволяет клиентам ознакомиться с людьми, которыми он будет взаимодействовать через ПМ.

На рисунке 47 представлен экран клиенты.

ID	Имя	Фамилия	Отчество	Телефон	Активность	
bf05ed73-b141-4ee4-8b92-e564d3129849	2	2	2	2	true	Изменить активность
15a9ceb1-5c7c-46ee-b1a5-35774a918182	client1	client1	client1	client1	true	Изменить активность
36a3cc2d-193f-4522-8c90-5a834ba7a786	client3	client3	client3	client3	true	Изменить активность
56395f49-f505-47d6-9d47-f493ffe71a5e	client2	client2	client2	client2	false	Изменить активность

Создание клиента:

Имя клиента

Фамилия клиента

Отчество клиента

телефон клиента

пароль клиента

Организация клиента

Рисунок 47 - Экран клиенты

Данный экран предоставляет возможность просматривать клиентов, изменять их активность что позволяет закрыть доступ к ПМ и создавать новых клиентов в ПМ.

На рисунке 48 представлен экран организации.

Фильтр:

Активные записи ☒

Поиск

ID	name	active	button
2	СМО Самара	true	Изменить активность
1	ТФОМС Самара	true	Изменить активность

Создание организации:

Имя организации

Сохранить

Рисунок 48 - Экран организации

Данный экран позволяет создавать организации и изменять их активность, если клиент находится в неактивной организации у него пропадает доступ к ПМ.

3.3. Тестирование ПМ ведения учетных записей медицинского центра

Для проверки функционала нашего продукта будет произведено ручное тестирование.

Ручное тестирование часть процесса тестирования на этапе контроля качества в процессе разработки программного обеспечения. Оно производится тестировщиком без использования программных средств, для проверки программы или сайта путём моделирования действий пользователя [14]. В роли тестировщиков могут выступать и обычные пользователи, сообщая исполнителям о найденных ошибках.

При открытии ПМ пользователь обязан авторизоваться что бы получить доступ к ПМ поэтому для начала проверим возможно перейти на другие ссылки через браузер без авторизации.

Для этого необходимо ввести существующие пути в адресную строку браузера, например, <http://localhost:8081/task>.

После этого должен произойти переход на адрес <http://localhost:8081/auth>

Данный функционал работает правильно и дает зайти к закрытым страницам без авторизации.

Далее необходимо проверить форму авторизации.

Попробуем ввести не правильный логин и пароль результат представлен ниже на рисунке 49.

!

Не верно указан номер или пароль

×

Форма авторизации

Номер телефона89371479157

Пароль.....

Разработчик☐

Войти

Рисунок 49 – Результат неверно введенного пароля

После чего попробуем ввести логин и пароль существующего исполнителя результат представлен на рисунке 50.

КлиентыИсполнителиЗадачиОрганизацияВыход

Фильтр задач:
Статус задачи
Исполнитель
Выбрать клиента
Поиск

ID	Тема	Дата задачи	Дата завершения	Клиент	Исполнитель	Статус	
36	Задача тест1	2025-04-26		2		Создана	Открыть заявку
37	Задача тест2	2025-04-26		2		Отмененна	Открыть заявку

Рисунок 50 – Результат верно введенного пароля

Функционал выполненлся как ожидалось, дав доступ пользователю в ПМ.

Далее перейдем в вкладку организация и попробуем создать новую организацию. Для этого пользователь должен являться исполнителем и на странице организации заполнить форму создать организацию и нажать кнопку сохранить.

Результат представлен ниже на рисунке 51.

ID	name	active	button
11	ТФОМС Самара	true	<button>Изменить активность</button>

Создание организации:

Имя организации

Сохранить

Рисунок 51 – Результат добавлении организации

Данный функционал правильно отработал после нажатия кнопки произошел вызов запроса на серверную часть, которая создала в таблице организации.

Далее проверим работоспособность функционала добавление новый клиентов. Для этого необходимо перейти на вкладку клиенты и заполнить форму и нажать кнопку сохранить. Результат представлен ниже на рисунке 52.

Клиенты
Исполнители
Задачи
Организация

Выход

ID	Имя	Фамилия	Отчество	Телефон	Активность	
b0269d5a-eac6-443c-b7a0-8f31cbe473ba	Тансия	Шеломова	Васильевна	+7 (935) 263-68-23	true	<button>Изменить активность</button>

Создание клиента:

Имя клиента

Фамилия клиента

Отчество клиента

телефон клиента

пароль клиента

Организация клиента

Сохранить

Рисунок 52 – Результат создание клиента

Данный функционал правильно отработал после нажатия кнопки произошел вызов запроса на серверную часть, которая создала в таблице клиент.

Далее проверим возможно выхода из ПМ, для этого необходимо нажать на шапке кнопку.

Результат нажатия кнопки выход представлен на рисунке 53.

Рисунок 53- Результат нажатие на кнопку выход

Как и ожидалось ПМ открыл форму авторизации.

Далее необходимо проверить что созданный выше клиент имеет возможность авторизоваться в ПМ.

Для этого необходимо ввести данные авторизации клиента и нажать на кнопку Войти.

Результат представлен на рисунке 54.

Рисунок 54 – Результат авторизации пользователя

По результату выяснилось, что клиент смог авторизоваться в ПМ. Также на основе результата выполнена проверка что клиенту закрыты доступы к созданию новых клиентов и организации так как данные операции могут выполнять только исполнители.

Далее проверим функционал создания задачи.

Для этого необходимо на вкладке задачи заполнить форму задачи.

Результат представлен на рисунке 55.

ID	Тема	Дата задачи	Дата завершения	Клиент	Исполнитель	Статус	
38	Доработка СЭД	2025-05-01		Тансия	admin	Отмененна	Открыть задачу

Рисунок 55 – Результат добавления задачи

Далее необходимо авторизоваться под исполнителем и проверить возможность взять задачу в работу.

Для этого необходимо нажать кнопку открыть задачу и в открытом модальном окне нажать на кнопку взять в работу.

На рисунке 56 представлено модальное окно задачи.

Задача: #38

Тема задачиДоработка СЭД

Сообщение задачи

Необходимо добавить функционал

1
2...

Исполнительadmin

Изменить задачу

К проверке задачуВзять в работуОтменить задачу

Рисунок 56 – Модальное окно задачи

После нажатие кнопки взять в работу в таблице измениться статус задачи.

Результат представлен на рисунке 57.

ID	Тема	Дата задачи	Дата завершения	Клиент	Исполнитель	Статус	
38	Доработка СЭД	2025-05-01		Тансия	admin	В работе	Открыть задачу

Рисунок 57 – Результат изменения статус задачи на значение в работе.

Далее после выполнения задачи исполнитель должен изменить статус на в проверке, результат представлен на рисунке 58.

ID	Тема	Дата задачи	Дата завершения	Клиент	Исполнитель	Статус	
38	Доработка СЭД	2025-05-01		Тансия	admin	В проверке	Открыть задачу

Рисунок 58 – Результат изменения статус задачи на значение в проверке

После клиент проверяет задачу и меняет статус на значение выполнена.

Результат представлен на рисунке 59.

ID ▲	Тема ▼	Дата задачи ▼	Дата завершения ▼	Клиент ▼	Исполнитель ▼	Статус ▼	
38	Доработка СЭД	2025-05-01		Тансия	admin	Выполнена	Открыть задачу

Рисунок 59 – Результат изменения статус задачи на значение в работе

Данным ручным тестированием проверены основные функциональность ПМ.

В ходе ручного тестирования программного модуля были успешно реализованы функциональные и нефункциональные требования к ПМ.

В данной главе реализована база данных PostgreSQL в ПМ, серверная часть ПМ на node используя фреймворк Nest представляющая собой архитектуру REST, клиентская часть ПМ на node используя фреймворк Vue3 используя методологию компонентов, представлен интерфейс ПМ, выбран метод тестирования ПМ и выполнено ручное тестирование ПМ.

ЗАКЛЮЧЕНИЯ

В первой главе проведен анализ предприятия ООО «ИМЦ», определена организационная структура предприятия ООО «ИМЦ», разработаны диаграммы IDFE0 процесса обработки обращения клиентов, разработана модель ТО-ВЕ процесса обработки обращения клиентов с использованием ПМ, разработаны диаграммы DFD и IDEF3, разработана физическая модель ПМ, произведен анализ аналогов требуемого ПМ, произведен выбор разработки собственного ПО, определены этапы и задачи процесса разработки ПМ, определены ресурсы процесса разработки ПМ, определены затраты на разработку ПМ, разработано ТЗ для разработки ПМ.

В второй главе были определены сущности ПМ, определены задачи выполняющие ПМ, разработаны диаграммы вариантов использования ПМ, классов ПМ, последовательности ПМ, компонентов ПМ, состояния ПМ, выбраны языки и фреймворки разработки front-end и back-end приложения, а также система управления базой данных и программные средства для разработки веб приложения

В третьем главе реализована база данных PostgreSQL в ПМ, серверная часть ПМ на node используя фреймворк Nest представляющая собой архитектуру REST, клиентская часть ПМ на node используя фреймворк Vue3 используя методологию компонентов, представлен интерфейс ПМ, выбран метод тестирования ПМ и выполнено ручное тестирование ПМ.

В заключение стоит отметить, что разработанный программный модуль является централизованным хранилищем данных которое позволило унифицировать и систематизировать работу обращениями клиента, это упростило и ускорило деятельность отдела технической поддержки. Благодаря внедрению данного решения повысилась эффективность обработки запросов, улучшилось качество сервиса и увеличилась удовлетворённость клиентов.

Список используемых источников

1. Информационно-медицинский центр : официальный сайт. — Самара. — URL: <https://imc-s.ru> (дата обращения: 30.04.2025). — Текст : электронный.
2. Диаграммы потоков данных DFD. — URL: <https://vc.ru/u/165346-evgeniy-kazak/562998-dfdy-dlya-nachinayushchih> (дата обращения: 03.05.2025). — Текст : электронный.
3. IDEF1X Моделирование данных. — URL: <https://proproprogs.ru/index.php/programs/analysis/92-idef1x-data-modeling> (дата обращения: 03.05.2025). — Текст : электронный.
4. IDEF3 — методология моделирования процессов. — URL: <https://sibac.info/idef3-metodologiya-modelirovaniya-protssesov/> (дата обращения: 03.05.2025). — Текст : электронный.
5. Методология IDEF0: что это такое, основные понятия, этапы построения. — URL: <https://habr.com/ru/companies/otus/articles/471674/> (дата обращения: 03.05.2025). — Текст : электронный.
6. Хатунцев, В. А. Современные системы управления проектами: Jira, Trello, Asana, YouTrack / В. А. Хатунцев. — Москва : ДМК Пресс, 2023. — 298 с. — ISBN 978-5-94074-564-8. — Текст : непосредственный.
7. Фаулер, М. UML. Основы / М. Фаулер, К. Скотт. — 3-е изд. — Санкт-Петербург : Символ-Плюс, 2018. — 192 с. — ISBN 978-5-93286-060-7. — Текст : непосредственный.
8. Структурный и объектно-ориентированный подходы к проектированию ИС. — URL: https://studme.org/210387/informatika/strukturnyy_obektno_orientirovannyu_podhod_u_proektirovaniyu (дата обращения: 03.05.2025). — Текст : электронный.
9. Немцева, Т. И. Компаративный анализ языков программирования для веб-разработки / Т. И. Немцева, К. В. Сидоров. — Санкт-Петербург : БХВ-Петербург, 2024. — 368 с. — ISBN 978-5-9775-6589-0. — Текст : непосредственный.

10. Сравнение Vue.js, React и Angular: выбор фреймворка. — URL: <https://habr.com/ru/articles/726490/> (дата обращения: 03.05.2025). — Текст : электронный.

11. VS Code vs WebStorm: какую IDE выбрать для веб-разработки. — URL: <https://medium.com/better-programming/vs-code-vs-webstorm-ide-for-web-development-e251dad93fea> (дата обращения: 03.05.2025). — Текст : электронный.

12. Сравнение PostgreSQL, MySQL и Microsoft SQL Server: что выбрать в 2025 году. — URL: <https://proglab.io/p/sravnenie-postgresql-mysql-i-ms-sql-server> (дата обращения: 03.05.2025). — Текст : электронный.

13. NestJS vs Django vs Spring Boot: сравнительный анализ back-end фреймворков. — URL: <https://devby.io/news/nestjs-vs-django-vs-spring-boot> (дата обращения: 03.05.2025). — Текст : электронный.

14. Канер, С. Тестирование программного обеспечения / С. Канер, Дж. Фолк, Е. Нгуен. — Москва : Диалектика, 2019. — 320 с. — ISBN 978-5-8459-2094-0. — Текст : непосредственный.

15. 1. Варламова, Л. Н. Управление документацией : англо-русский аннотированный словарь стандартизированной терминологии / Л. Н. Варламова, Л. С. Баюн, К. А. Бастрикова. — Москва: Спутник+, 2017. — 398 с . ; 21 см. — Библиогр.: с. 358—360. — 100 экз. — ISBN 978-5-9973-4489-4. — Текст: непосредственный.

16. Шлеин, В. А. Основы менеджмента : учебное пособие / В. А. Шлеин, Е. А. Иванова. — Москва : РУТ (МИИТ), 2020. — 135 с. — Текст : электронный // Лань: электронно-библиотечная система. — URL: <https://e.lanbook.com/book/175838> (дата обращения: 27.05.2025). — Режим доступа: для авториз. пользователей.

17. Российская Федерация. Законы. Об общих принципах организации местного самоуправления в Российской Федерации : Федеральный закон № 131-ФЗ : [принят Государственной думой 16 сентября 2003 года : одобрен Советом Федерации 24 сентября 2003 года]. — Москва : Проспект ; Санкт-Петербург :

Кодекс, 2017. — 158 с. ; 20 см. — 1000 экз. — ISBN 978-5-392-26365-3. — Текст : непосредственный.

18. Российская Федерация. Законы. Уголовный кодекс Российской Федерации : У К : текст с изменениями и дополнениями на 1 августа 2017 года : [принят Государственной думой 24 мая 1996 года : одобрен Советом Федерации 5 июня 1996 года]. — Москва : Эксмо, 2017. — 350 с . ; 20 см. — (Актуальное законодательство). — 3000 экз. — ISBN 978-5-04-004029-2. — Текст : непосредственный.

19. Правительство Российской Федерации : официальный сайт. — Москва. — Обновляется в течение суток. — URL: <http://government.ru> (дата обращения: 20.05.2025). — Текст : электронный.

20. Государственный Эрмитаж : [сайт]. — Санкт-Петербург, 1998 — URL: <http://www.hermitagemuseum.org/wps/portal/hermitage> (дата обращения: 23.05.2025). — Текст. Изображение : электронные.

21. ТАСС : информационное агентство России : [сайт]. — Москва, 1999 — Обновляется в течение суток. — URL: <http://tass.ru> (дата обращения: 27.05.2025). — Текст : электронный.

22. Грязев, А. Пустое занятие: кто лишает Россию права вето в СБ ООН: в ГА ООН возобновлены переговоры по реформе Совета Безопасности / А. Грязев. — Текст : электронный // Газета.ru : [сайт]. — 2018. — 2 февр. — URL: https://www.gazeta.ru/politics/2018/02/02_a_11634385.shtml (дата обращения: 27.05.2025).

23. План мероприятий по повышению эффективности госпрограммы «Доступная среда». — Текст: электронный // Министерство труда и социальной защиты Российской Федерации: официальный сайт. — 2017. — URL: <https://rosmintrud.ru/docs/1281> (дата обращения: 27.05.2025)