

Министерство науки и высшего образования Российской Федерации



Санкт-Петербургский горный университет

Кафедра информационных систем и вычислительной техники

Допускается к защите в ГЭК

Заведующий кафедрой ИСиВТ

доцент _____ /Мазаков Е.Б./
(подпись)

“ ” _____ 2022 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

(дипломный проект бакалавра)

на тему: *«Разработка CRM-системы для организации консультаций
в медучреждениях»*

Направление подготовки 09.03.01 - Информатика и вычислительная техника

(шифр)

(наименование направления)

Направленность (профиль)

информации и управления

Автоматизированные системы обработки

Автор: студент гр. ИАС-18 180271

(шифр)

_____ /Белокон Ю.А./

(подпись)

(Ф.И.О.)

Руководитель: _____

(должность, звание)

_____ /Гурко А.В. /

(подпись)

(Ф.И.О.)

Рецензент: _____

(должность, звание)

_____ / _____ /

(подпись)

(Ф.И.О.)

Санкт-Петербург

2022

**ПЕРВОЕ ВЫСШЕЕ ТЕХНИЧЕСКОЕ УЧЕБНОЕ ЗАВЕДЕНИЕ
РОССИИ**



**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**федеральное государственное бюджетное образовательное учреждение высшего
образования**

«САНКТ-ПЕТЕРБУРГСКИЙ ГОРНЫЙ УНИВЕРСИТЕТ»

Кафедра Информационных систем и вычислительной техники

УТВЕРЖДАЮ

Заведующий кафедрой

_____ **Мазаков Е.Б.**

« ____ » _____ **2021г.**

ЗАДАНИЕ

**НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА**

студенту Белокон Ю.А. ИАС-18 180271

(Ф.И.О.)

(шифр группы)

(индекс)

Тема: *Разработка CRM-системы для организации консультаций в
медучреждениях*

Исходные данные:

методика построения базы данных, инструменты разработки – HTML, SQL, PHP,
CSS, JavaScript

Тема специальной части:

разработка Web-приложения и базы данных

*Требования к графической части проекта и пояснительной записке содержатся в
Методических указаниях по дипломному проектированию.*

Задание выдал (Руководитель ВКР) _____ / Гурко А.В. /

(должность, подпись) (Ф.И.О.)

Задание принял к исполнению студент _____ / Белокон Ю.А. /

(подпись)

(Ф.И.О.)

Дата получения задания: 20 октября 2021г.

Аннотация

В данной выпускной квалификационной работе представлена разработка CRM-системы для организации консультаций в медицинских учреждениях. Система содержит полный набор функций, позволяющий автоматизировать соответствующие процессы организации коммуникации пациента и медучреждения.

Выпускная квалификационная работа состоит из введения, четырех глав, восьми приложений, заключения, перечня использованной литературы.

В первой главе описывается предметная область, рассматриваются вопросы актуальности разработки и внедрения описанной системы на предприятии. Обоснован выбор способов проектирования и средств программной реализации.

Второй раздел содержит описание процесса разработки базы данных, структуры приложения и его интерфейса.

Третий раздел содержит описание функционала приложения и варианты взаимодействия с ним при помощи интерфейса.

Четвертый раздел представляет технико-экономические расчеты. Определяются расходы на внедрение и сопровождение, дается экономическое обоснование эффективности проекта.

Выпускная квалификационная работа содержит 70 страниц, 41 рисунок, 5 таблиц, 10 формул, 8 приложений и список использованной литературы.

Abstract

This graduation paper presents the development of a CRM system for organizing consultations in medical institutions. The system contains a full set of functions that allows you to automate the relevant processes of organizing communication between the patient and the medical institution.

The graduation paper consists of an introduction, four chapters, eight appendices, a conclusion, a list of references.

The first chapter describes the subject area, discusses the relevance of the development and implementation of the described system in the enterprise. Also the choice of design methods and software implementation tools is substantiated in this chapter.

The second section contains a description of the database development process, the structure of the application and its interface.

The third section contains a description of the application's functionality and options for interacting with the applying of the interface.

The fourth section presents technical and economic calculations. The costs of implementation and maintenance are determined, and an economic justification for the effectiveness of the project is given in this chapter.

The graduation paper contains 70 pages, 41 figures, 5 tables, 10 formulas, 8 appendices and a list of references.

Содержание

Аннотация	3
Abstract	4
Введение.....	7
Раздел 1. Аналитическая часть.....	9
1.1 Анализ предметной области, определение задачи и требований	9
1.2 Анализ функций существующих аналогов.....	10
1.2.1 Медицинская информационная система МЕДИАЛОГ	10
1.2.2 БИТ.Управление медицинским центром	10
1.2.3 ТрастМед:МИС SaaS.....	10
1.3 Актуальность разработки.....	11
1.4 Выбор технологий и средств разработки.....	12
Выводы.....	13
Раздел 2. Практическая часть.....	14
2.1 Возможности приложения	14
2.2 Проектирование базы данных.....	15
2.2.1 Назначение и функции базы данных	16
2.2.2 Инфологическое проектирование	17
2.2.3 Концептуальное проектирование.....	22
2.2.4 Физическое проектирование.....	25
2.3 Разработка веб-приложения.....	30
2.3.1 Разработка Frontend-части	31
2.3.2 Разработка Backend-части.....	32
Выводы.....	34
Раздел 3. Пример работы приложения	35
3.1 Запуск приложения. Главная страница.....	35
3.2 Возможности личного кабинета пользователя с ролью «администратор».....	36
3.3 Возможности личного кабинета пользователя с ролью «врач».....	38

3.4 Возможности личного кабинета пользователя с ролью «пациент»	40
Раздел 4. Экономическая часть	46
4.1 Оценка инвестиционных расходов	46
4.2 Оценка эксплуатационных расходов	49
4.3 Оценка эффективности проекта.....	51
Выводы.....	52
Заключение.....	53
Список использованной литературы.....	55
Приложение 1	56
Приложение 2	59
Приложение 3	61
Приложение 4	65
Приложение 5	66
Приложение 6	68
Приложение 7	69
Приложение 8	70

Введение

В наше время наблюдается активное развитие информационных технологий, призванное максимально автоматизировать бизнес-процессы для более эффективной работы предприятий. Также большое внимание уделяется внедрению различных программных продуктов для оптимизации работы медицинских учреждений, решения множества задач в процессе предоставления медицинских услуг. Именно это явилось побуждением к написанию данной работы.

Итак, цель работы состоит в разработке CRM-системы для организации консультаций в медучреждениях. Несомненно, уже существуют программные решения для медицинских учреждений с похожей целью. Однако, изучив их, можно заметить, что каждому из них не хватает некоторого функционала. В данной работе представлено авторское видение максимально полного функционала и его реализация в формате веб-приложения.

Опираясь на поставленную цель, можно выделить этапы, требуемые для ее достижения:

- 1) анализ предметной области;
- 2) обоснование необходимости создания CRM-системы для организации консультаций в медучреждениях;
- 3) формулировка требований к проектируемой системе;
- 4) выбор технологий и инструментальных средств разработки;
- 5) выбор модели данных;
- 6) проектирование базы данных для CRM-системы;
- 7) проектирование внешней модели;
- 8) оценка экономической эффективности проекта.

Объектом исследования является автоматизация процесса записи на прием в медицинское учреждение.

Предметом исследования является программное обеспечение для автоматизации процесса записи пациента на прием в медицинское учреждение.

К методам решения поставленных задач относится анализ существующих в настоящее время аналогов, решающих поставленную в данной выпускной квалификационной работе проблему, и теоретический анализ предметной области.

Структура выпускной квалификационной работы представляет собой введение, 4 раздела, заключение, список использованной литературы из 13 наименований и 8 приложений.

Первый раздел содержит теоретический обзор предметной области и обоснование выбора технических инструментов проектирования.

Второй раздел содержит описание процесса разработки базы данных, структуры приложения и его интерфейса.

Третий раздел содержит описание функционала приложения и варианты взаимодействия с ним при помощи интерфейса.

Четвертый раздел содержит экономическое обоснование эффективности проекта.

Выпускная квалификационная работа содержит 70 страниц, 41 рисунок, 5 таблиц, 10 формул, 8 приложений и список использованной литературы.

Источниками для написания послужили работы отечественных и зарубежных авторов: учебная и учебно-методическая литература, литература демонстрационных версий программных продуктов, а также ресурсы сети Интернет.

Раздел 1. Аналитическая часть

Данный раздел содержит обоснование актуальности разработки, обзор существующих аналогов, формулировка технического задания, выбор технологий, средств и среды разработки, обоснование выбора.

1.1 Анализ предметной области, определение задачи и требований

Перед постановкой задачи разработки было необходимо изучить, осмыслить и проанализировать предметную область, чтобы определить требования к функциональности и границы проекта. Предметная область сильно влияет на все аспекты проекта: требования к системе, взаимодействие с пользователем, модель хранения данных, реализацию и так далее.

Идея разработки, проведенной в рамках данной выпускной квалификационной работы, состоит в том, чтобы создать удобное интуитивно понятное веб-приложение, позволяющее пользователям в любой момент времени, имея подключение к сети Интернет, связываться с медицинским учреждением. При этом пользователь может быть как врачом, так и пациентом.

В результате опроса пользователей ныне существующих аналогичных продуктов был получен список функций, необходимых к реализации для достижения описанной выше цели:

- 1) три группы пользователей: администратор системы, врач, пациент;
- 2) возможность хранения в профиле пользователя личной информации для его идентификации;
- 3) возможность быстрого определения свободных мест в расписании приема врача и записи пациента на прием ко врачу (самостоятельной и при помощи администратора);
- 4) опция просмотра врачом списка записанных на прием пациентов;
- 5) возможность ведения электронной медицинской карты: врачу нужен доступ с правами редактирования, пациенту – доступ с правами чтения;
- 6) функция отправки жалобы в медицинское учреждение – для пациентов, функция просмотра жалоб – для администратора;
- 7) опция отправки сообщения пациентом врачу на почту из приложения.

Таким образом, определены необходимые функции CRM-системы для организации консультаций в медучреждениях и определены границы проекта.

1.2 Анализ функций существующих аналогов

Следующим этапом стал анализ функций существующих аналогов, который немаловажен при определении актуальности проекта.

1.2.1 Медицинская информационная система МЕДИАЛОГ

МИС МЕДИАЛОГ является одним из лидеров рынка медицинских информационных систем. С ее помощью автоматизировано более 950 медицинских учреждений и свыше 25 000 рабочих мест в России и СНГ. Этим продуктом активно пользуется сеть клиник «МЕДСИ», перинатальный центр и сеть клиник «Мать и дитя», ГКБ им. С.П. Боткина (Москва), поликлиники ОАО «Газпром» и прочие [7].

Внедренная МИС МЕДИАЛОГ представляет собой стационарное приложение, устанавливаемое на компьютерах медицинской организации. При этом на веб-сайте добавляется раздел-интеграция с этим приложением для возможности записи на прием в режиме онлайн. В приложении предусмотрена возможность просмотра врачом расписания собственного приема и ведения электронных медицинских карт. Существует опция экспорта ЭМК в формате HTML, предусмотрены различные шаблоны медкарт. Интерфейс приложения выглядит довольно просто, даже наверно не очень современно.

1.2.2 БИТ.Управление медицинским центром

БИТ.Управление медицинским центром – разработка компании 1С, широко используемая в настоящее время. В данный комплекс программных средств входит медицинская CRM-система, изученная в ходе выполнения настоящей ВКР. CRM-система включает механизм для работы с электронными медицинскими картами, историями болезни и результатами анализов, анализа истории посещений клиента, имеются готовые шаблоны приема для врачей разных должностей, электронная цифровая подпись. Интерфейс выглядит современно, удобен в использовании, интуитивно понятен [8].

1.2.3 ТрастМед:МИС SaaS

ТрастМед:МИС SaaS представляет собой комплекс средств автоматизации деятельности персонала лечебно-профилактических учреждений. В комплекс входит ТрастМед:Поликлиника, используемая для автоматизации деятельности персонала амбулаторно-поликлинического подразделения, и ТрастМедСтационар, задача которого – автоматизировать деятельность персонала стационара [9].

Программный комплекс ТрастМед:МИС SaaS выполняет следующие функциональные задачи:

- 1) создание и ведение медицинской истории;
- 2) регистрация отказа от/в госпитализации;
- 3) распределение пациентов в отделения стационара на лечение, возможность перевода между стационарами;
- 4) назначение/смена лечащего врача пациента;
- 5) учет установленных диагнозов;
- 6) учет услуг, оказанных пациенту во время лечения в стационаре;
- 7) формирование и печать выписки пациента из стационара;

1.3 Актуальность разработки

В результате изучения нескольких продуктов-лидеров рынка и сравнения с ними можно отметить несколько ключевых моментов, которые возможно улучшить.

- 1) Сложность внедрения и поддержки. В данной работе представлен вариант реализации CRM-системы в виде веб-приложения. Никакие прочие установки не требуются. Также можно легко совместить результат работы с сайтом медицинского учреждения.
- 2) Уход от необходимости услуги «звонок от врача». Эта услуга требует больших затрат рабочего времени врача, а также налаживания одновременного контакта врача и пациента. В работе предлагается организовать общение врача и пациента посредством переписки по электронной почте. Таким образом врач будет получать структурированное сообщение от пациента и отвечать на него в удобное время в течение рабочего дня. Пациент же также будет иметь возможность написать врачу в любое время, останется только дождаться ответа. Такая схема является более удобной и менее времязатратной.
- 3) Доступность системы вне медучреждения. В рассмотренных медицинских информационных системах врачу не предоставлена возможность доступа к системе за пределами рабочего места. Он не может доделывать рабочие дела дома, приходится задерживаться на работе.
- 4) Экономия бумажных носителей. Если пациент будет иметь доступ к результатам собственных визитов ко врачу отовсюду и в любое время, не будут требоваться выписки с рекомендациями на бумажных носителях.

- 5) Обратная связь. Для анализа качества сервиса в медицинском учреждении является удобным обеспечить пациентам возможность быстро и легко оставить отзыв о визите к конкретному врачу.

Таким образом, разработка CRM-системы для организации консультаций в медучреждениях является актуальной.

1.4 Выбор технологий и средств разработки

Очень важным этапом процесса разработки является определение программных средств для реализации поставленных задач и обеспечения выполнения предъявляемых требований.

В данной выпускной квалификационной работе представлена разработка веб-приложения при помощи языка разметки HTML, языка стилей CSS, языков программирования PHP и JavaScript. Для работы с базой данных в проекте использована классическая СУБД – MySQL, а локальный сервер для БД развернут средствами Localhost Denwer.

HTML является единственным языком разметки, на котором написана разметка всех веб-сайтов. Этот язык позволяет веб-браузеру понять, как строить интерфейс страницы. HTML-код хранится на сервере, веб-браузер получает его и интерпретирует в интерфейс.

HTML позволяет построить основную схему интерфейса, но изначальный (стандартный) внешний вид этой схемы оставляет желать лучшего. Здесь начинается работа языка стилей CSS: он дает разработчику возможность установить кастомный внешний вид любым элементам веб-страницы.

После создания frontend-части необходимо разработать backend-часть, то есть настроить обработку данных, которые приложение будет получать при взаимодействии пользователя с интерфейсом. В данной ВКР использован язык PHP для обработки обращений к базе данных, анализа их результатов, а также проведения всех необходимых операций с данными, JavaScript – для более гибкой работы интерфейса.

MySQL является стандартной СУБД, получившей широкое использование. Отличается очень высокой скоростью выполнения запросов. Имеет возможность кэширования запросов, что значительно ускоряет процесс обработки запросов для сайтов, на которых преобладают неоднократно повторяющиеся запросы на чтение.

Для разработки и отладки работы веб-приложения и связи с БД очень удобно использовать Localhost Denwer, который позволяет развернуть виртуальный локальный сервер на персональном компьютере. Позволяет быстро и легко проверить работоспособность программного продукта перед введением его в эксплуатацию.

Выводы

В аналитической части данной выпускной квалификационной работы произведен анализ предметной области, формулировка задачи на разработку и определение требований к проекту. Изучены медицинские информационные системы, занимающие лидирующие позиции на рынке в настоящее время и являющиеся аналогами разработанной в рамках данной ВКР CRM-системы.

Обоснован выбор технологии и средств разработки: HTML, CSS – frontend, PHP, JavaScript – backend, MySQL – СУБД, Localhost Denwer – средство создания виртуального локального сервера.

Раздел 2. Практическая часть

В данном разделе будут рассмотрены этапы разработки веб-приложения: выбор СУБД, создание БД, разработка интерфейса приложения, подключение БД к проекту, настройка элементов взаимодействия пользователя с интерфейсом, организация выполнения операций с данными.

2.1 Возможности приложения

Структура CRM-системы для организации консультаций в медицинских учреждениях состоит из трех частей, поскольку предусмотрено три типа пользователей, каждому из которых доступны разные действия. На рис. 2.1 представлена структура для пользователя с ролью «администратор», на рис. 2.2 – для врача, на рис. 2.3 – для пациента.



Рисунок 2.1 Возможности пользователя с ролью «администратор»



Рисунок 2.2 Возможности пользователя с ролью «врач»



Рисунок 2.3 Возможности пользователя с ролью «пациент»

2.2 Проектирование базы данных

Проектирование реляционной базы данных – процесс, состоящий из четырех этапов:

- 1) инфологическое проектирование;
- 2) концептуальное проектирование;
- 3) физическое проектирование;
- 4) проектирование внешней модели.

На каждом из перечисленных этапов формируется модель данных – схема, представленная на языке описания каждого уровня.

Прежде всего анализируется предметная область, определяется назначение базы данных и строится инфологическая модель, дающая общее представление о выбранной области. В этом и заключается первый этап проектирования БД. Инфологическая модель данных не зависит от типа выбранной целевой СУБД, языка реализации, набора создаваемых прикладных программ, а также от любых других

компонентов физической реализации. Такая модель описывается на естественном языке, также можно использовать формулы, таблицы, графики, диаграммы.

Вторым этапом проектирования реляционной БД является преобразование имеющейся инфологической модели данных в концептуальную. Для этого необходимо выделить набор зависимостей, правил для адекватного отображения предметной области с учетом выбранной целевой СУБД.

Следующий этап – уже физическое проектирование. На этом этапе проектировщик базы данных принимает решения о способах реализации БД.

Завершающим этапом является проектирование внешней модели – модели, отображающей информацию о предметной области для прикладных программ и пользователей системы. Такая модель позволяет программам и пользователям системы получать доступ к данным по именам, не заботясь об их физическом расположении. В данной работе внешней моделью БД является веб-приложение.

2.2.1 Назначение и функции базы данных

В рамках приложения «CRM-система для организации консультаций в медицинских учреждениях» спроектирована база данных, обладающая следующими функциями:

- 1) добавление новых пользователей в систему – функция, доступная только администраторам системы, позволяет добавлять новых администраторов, врачей и пациентов;
- 2) редактирование данных профиля пользователей – каждый пользователь должен иметь набор данных, необходимых для его идентификации, в личном кабинете, а в случае необходимости должна быть возможность изменить эти данные;
- 3) создание записей в таблице учета посещений – единый список записей пациентов ко врачам, врачам доступно только чтение, а пациентам – чтение и запись, при этом присутствует фильтрация: врачу демонстрируются только записи к нему же, а пациенту – только его собственные записи;
- 4) создание и редактирование данных медкарт по результатам визитов – результаты посещения врача пациентом хранятся в удобном виде и доступны пациентам для чтения, а врачам для записи, редактирования и чтения;

- 5) добавление записей о жалобах пациентов – функция, полезная при анализе качества предоставляемого пациентам сервиса, доступна пациентам для записи, администраторам – для чтения.

2.2.2 Инфологическое проектирование

Инфологическая или информационно-логическая модель – это модель предметной области, определяющая информационные объекты, их атрибуты, отношения между объектами, динамику изменения предметной области, характер информационных потребностей пользователей. Эта модель применяется после словесного описания предметной области, то есть после постановки задачи, и не зависит от типа СУБД.

Инфологическое проектирование прежде всего связано с попыткой представления семантики (смыслового содержания) предметной области в модели базы данных. Реляционная модель данных в силу своей лаконичности не позволяет отобразить семантику. Существует несколько семантических моделей, но в настоящее время фактическим стандартом является модель «сущность-связь» или ER-модель («Entity Relationship»), предложенная Ченом в 1976 году.

В основе ER-модели лежат понятия «сущность» и «связь». Сущность имеет уникальное в пределах моделируемой системы имя. Объект, которому соответствует понятие сущности, имеет ряд атрибутов – характеристик, однозначно определяющих его свойства. Между сущностями могут быть установлены связи, показывающие соотношение сущностей. Связи могут быть нескольких типов: один к одному (1:1), один ко многим (1:M), многие к одному (M:1) и многие ко многим (M:M) [1].

В таблице 2.1 представлены сущности и их атрибуты для рассматриваемой предметной области.

Таблица 2.1. Сущности предметной области

Сущность	Описание	Атрибут
Пользователь	Хранит данные о пользователе	Уникальное имя пользователя
		Адрес электронной почты
		Номер телефона
		Имя

Сущность	Описание	Атрибут
		Фамилия
		Отчество
		Дата рождения
Роль пользователя	Хранит данные о ролях пользователей	Идентификатор
		Уникальное имя пользователя
		Роль
Данные регистрации	Хранит данные для авторизации пользователей	Идентификатор
		Уникальное имя пользователя
		Пароль (зашифрован)
Доктор	Хранит данные о врачах	Идентификатор
		Уникальное имя пользователя
		Должность
Посещение	Хранит данные о посещениях врачей	Логин пользователя
		Идентификатор посещения
		Дата посещения
		Время посещения
		Логин доктора
Карта	Хранит данные о медкартах	Идентификатор карты
		Логин пользователя
		Диагноз
		Рекомендации
		Дата
		Логин врача
Жалоба	Хранит записи о жалобах от пациентов	Идентификатор жалобы
		Логин пациента
		Логин врача
		Содержание жалобы

После обозначения сущностей и их атрибутов определяются связи между сущностями.

Связь «один к одному» (1:1) – в каждый момент времени каждому экземпляру одной сущности соответствует не более одного экземпляра другой сущности.

Связь «один ко многим» (1:M) – каждому экземпляру одной сущности (находящейся в связи слева) соответствуют 0, 1 или несколько представителей другой сущности.

Связь «многие к одному» (M:1): несколько экземпляров первой сущности (находящейся в связи слева) соответствует один экземпляр второй сущности.

Связь «многие ко многим» (M:M): каждому экземпляру одной сущности соответствуют 0, 1 или несколько представителей другой сущности и наоборот [2].

В таблице 2.2 показаны связи между имеющимися сущностями.

Таблица 2.2. Связи между сущностями

Связываемые сущности	Связь	Вид связи	Описание
Пользователь	В системе можно регистрировать новых пользователей	1:1	Один пользователь может иметь только один зарегистрированный аккаунт
Данные регистрации		1:1	
Доктор	Существует отдельный учет докторов в системе	1:1	Один врач имеет единственный аккаунт
Данные регистрации		1:1	
Роль	При регистрации новому пользователю назначается роль	1:M	Одному пользователю назначается только одна роль. Одна и та же роль может быть назначена нескольким людям
Данные регистрации		1:1	
Медицинская карта	По итогу посещения врач делает запись в медкарту пациента	1:1	Запись в медкарте пользователя принадлежит только ему. У пользователя может быть несколько записей в медкарте
Пользователь		1:M	
Посещение	Пациенты ходят на	M:1	Один пациент может

Связываемые сущности	Связь	Вид связи	Описание
Пользователь	прием ко врачам	1:M	неоднократно записываться на прием
Жалоба	Пациент может оставить жалобу на врача	M:1	На одного врача может быть подано несколько жалоб.
Пользователь		1:M	Одинаковые жалобы могут относиться к нескольким врачам
Медицинская карта	Доктор записывает результаты приема в медкарту	1:1	Один врач может заполнять несколько медкарт. Одна запись в медкарту пишется одним врачом
Доктор		1:M	
Посещение	Посещение всегда связано с соответствующим врачом	1:1	К одному врачу можно ходить несколько раз. Одно посещение относится к единственному врачу
Доктор		1:M	
Жалоба	Пациент может написать жалобу на врача	M:M	На одного врача можно написать несколько жалоб. Могут быть одинаковые жалобы на разных врачей
Доктор		1:M	
Медицинская карта	Медкарта заполняется за определенное посещение	1:1	За одно посещение может добавиться только одна запись в медкарту
Посещение		1:1	

После проведения описанного выше анализа имеется достаточно данных для построения инфологической модели. В данной работе представлена инфологическая модель базы данных в нотации Чена (ER-модель) (рисунок 2.4).

Условные обозначения ER-диаграммы:

- 1) «прямоугольник» - сущность предметной области;
- 2) «ромб» - связь между сущностью и атрибутом;
- 3) «овал» - атрибут сущности.

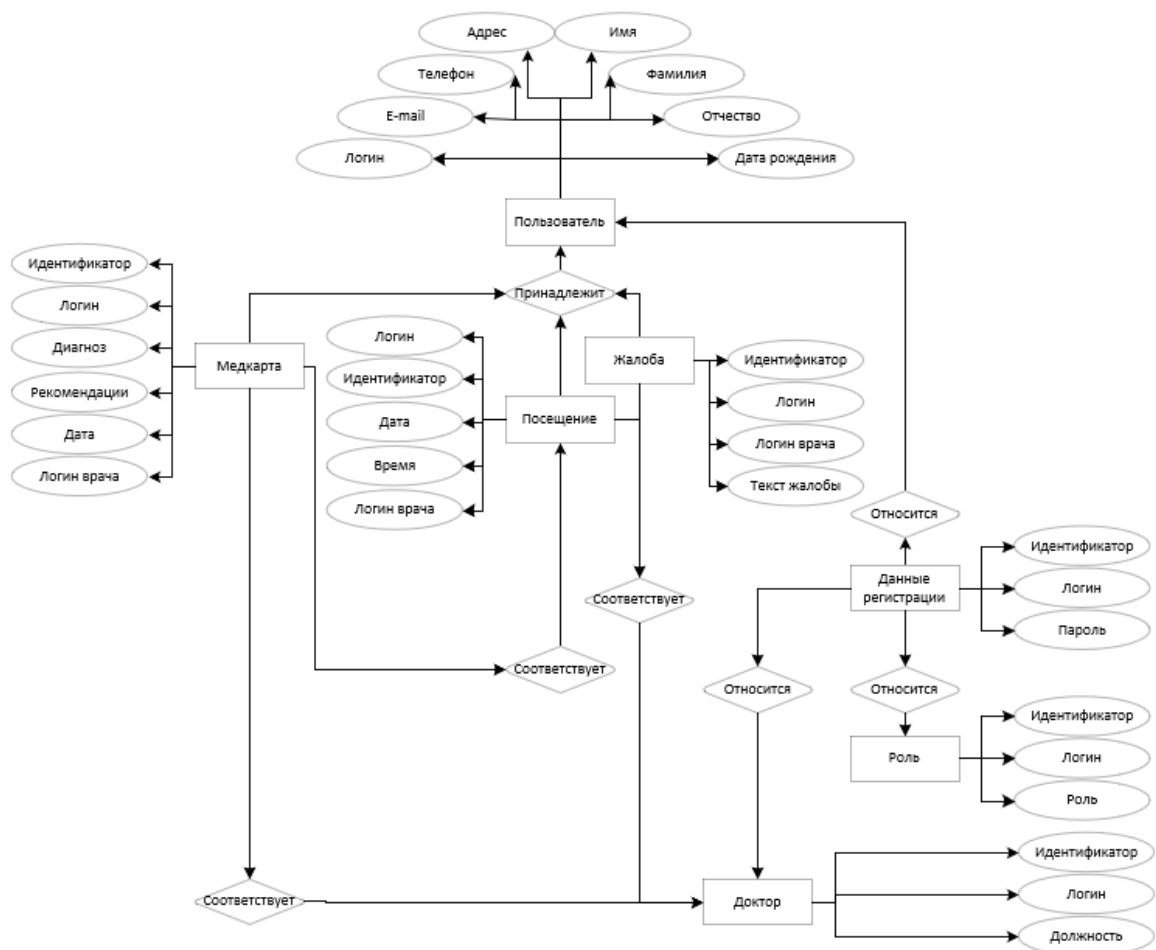
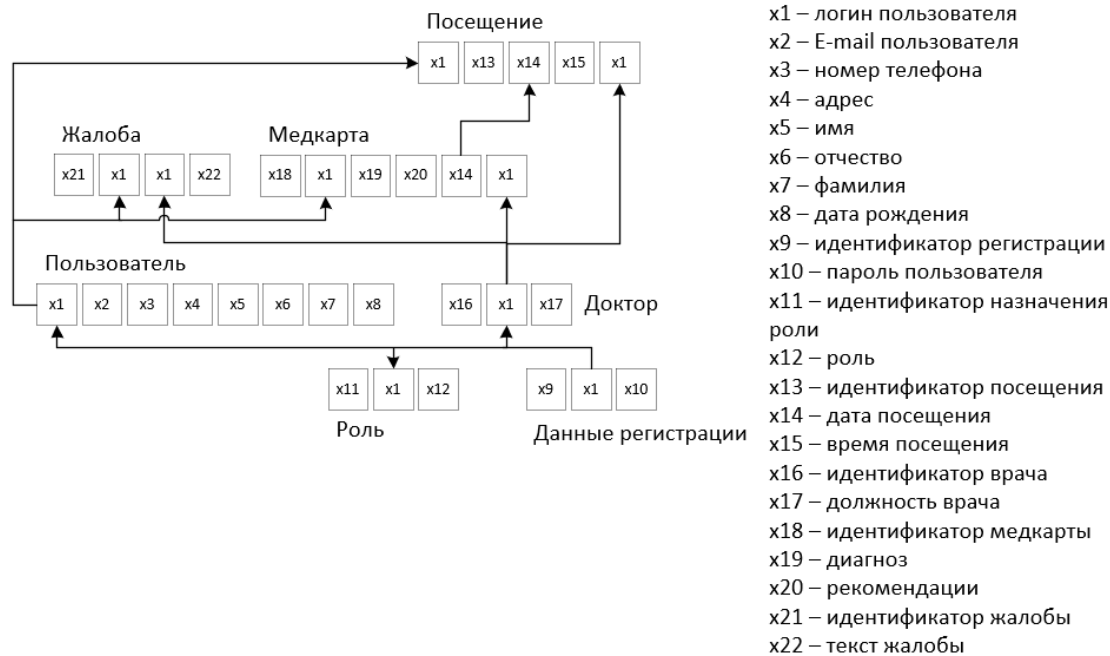


Рисунок 2.4 Инфологическая модель базы данных в нотации Чена

Все объекты и отношения могут быть представлены в другом виде, как показано на рисунке 2.5. Эта модель показывает отношения всех атрибутов модели.



2.2.3 Концептуальное проектирование

На концептуальном уровне реляционная модель представляется набором отношений. Структура этого набора определяется структурой функциональных, многозначных и других видов зависимостей, присущих данной предметной области.

Функциональная зависимость первого атрибута от второго состоит в том, что каждому значению свойства, описываемого вторым атрибутом, может быть поставлено в соответствие только одно значение свойства, описываемого первым атрибутом. Отношения должны адекватно отображать предметную область, выбранный состав отношений должен удовлетворять минимальной избыточности атрибутов, модель должна оставаться адекватной при modify-операциях над хранимой информацией.

Обозначения атрибутов схемы функциональной зависимости представлены в таблице 2.3.

Таблица 2.3. Связи между сущностями

Обозначение	Описание	Обозначение в схеме концептуальной модели
X1	Логин пользователя	username
X2	Адрес электронной почты	email
X3	Номер телефона	phone
X4	Адрес	address
X5	Имя	firstname
X6	Отчество	lastname
X7	Фамилия	surname
X8	Дата рождения	birthday
X9	Идентификатор регистрации	registration_id
X10	Пароль пользователя	password
X11	Идентификатор назначения роли	id
X12	Роль	role
X13	Идентификатор посещения	visit_id
X14	Дата посещения	date
X15	Время посещения	time

Обозначение	Описание	Обозначение в схеме концептуальной модели
X16	Идентификатор врача	doc_id
X17	Должность врача	doctype
X18	Идентификатор медкарты	card_id
X19	Диагноз	diagnosis
X20	Рекомендации врача	notes
X21	Идентификатор жалобы	complaint_id
X22	Текст жалобы	complaint

Функциональные зависимости:

1. $x1 \rightarrow x2x3x4x5x6x7x8x10x17x14x15x22$

Логин пользователя однозначно определяет адрес электронной почты, номер телефона, адрес, имя, отчество, фамилию, дату рождения, пароль пользователя, роль, дату посещения, время посещения, должность врача, текст жалобы.

2. $x1x14 \rightarrow x19x20$

Логин пользователя и дата посещения однозначно определяют диагноз и рекомендации врача.

По завершении вывода функциональных зависимостей можно определить ключ отношения. Ключ отношения – это некоторый атрибут или их совокупность, который однозначно определяет каждый кортеж отношения. На практике, правда, может быть несколько групп атрибутов, удовлетворяющих этому требованию.

Ключ отношений должен обладать идентифицируемостью и избыточностью. Это гарантирует то, что в отношении не будет двух кортежей с одинаковыми ключами, и то, что ключ будет состоять из минимального количества атрибутов.

Атрибут $x1$ является ключом отношения, поскольку соответствует требованиям уникальности и минимальности.

Следующим после определения ключа этапом является этап нормализации отношений, состоящий в приведении базы данных к нормальной форме. Для того,

чтобы БД можно было считать нормализованной, она должна быть приведена минимум к третьей нормальной форме.

- Первая нормальная форма. Переменная отношения находится в первой нормальной форме тогда и только тогда, когда в любом допустимом значении этой переменной отношения каждый ее кортеж содержит единственное значение для каждого атрибута.
- Вторая нормальная форма. Переменная отношения находится во второй нормальной форме тогда и только тогда, когда находится в первой нормальной форме и каждый атрибут, не являющийся ключом, неприводимо зависит от ее первичного ключа.
- Третья нормальная форма. Переменная отношения находится в третьей нормальной форме тогда и только тогда, когда находится во второй нормальной форме и ни один не являющийся ключом атрибут не является транзитивно зависимым от ее первичного ключа [3].

Схема отношения R:

$R(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}, x_{21}, x_{22})$

Исходя из вышеуказанных требований, получатся следующие отношения:

$R1(\underline{x_1}, x_2, x_3, x_4, x_5, x_6, x_7, x_8)$ образует таблицу users – таблицу данных профилей пользователей.

$R2(\underline{x_1}, \underline{x_9}, x_{10})$ образует таблицу registration – таблицу с данными для авторизации пользователей.

$R3(\underline{x_1}, \underline{x_{11}}, x_{12})$ образует таблицу roles – таблицу ролей пользователей.

$R4(\underline{x_1}, \underline{x_{13}}, \underline{x_{14}}, x_{15})$ образует таблицу visits – таблицу, хранящую записи на прием ко врачам.

$R5(\underline{x_1}, \underline{x_{16}}, x_{17})$ образует таблицу doctors – таблицу, содержащую перечень врачей и их должностей.

$R6(\underline{x_1}, \underline{x_{14}}, \underline{x_{18}}, x_{19}, x_{20})$ образует таблицу cards – таблицу, хранящую результаты приемов.

R7(x1, x21, x22) образует таблицу complaints – таблицу, в которую записываются все жалобы пациентов на качество сервиса, оказания медицинской помощи и прочие.

Здесь одной чертой подчеркнуты внешние ключи, двумя – первичные.

На рисунке 2.6 представлена логическая модель базы данных, разработанной в рамках данной выпускной квалификационной работы. Она была построена с помощью интерфейса Designer PHPMyAdmin, который позволяет визуальное управлять связями. Данная модель раскрывает структуру хранимых таблиц в системе MySQL.

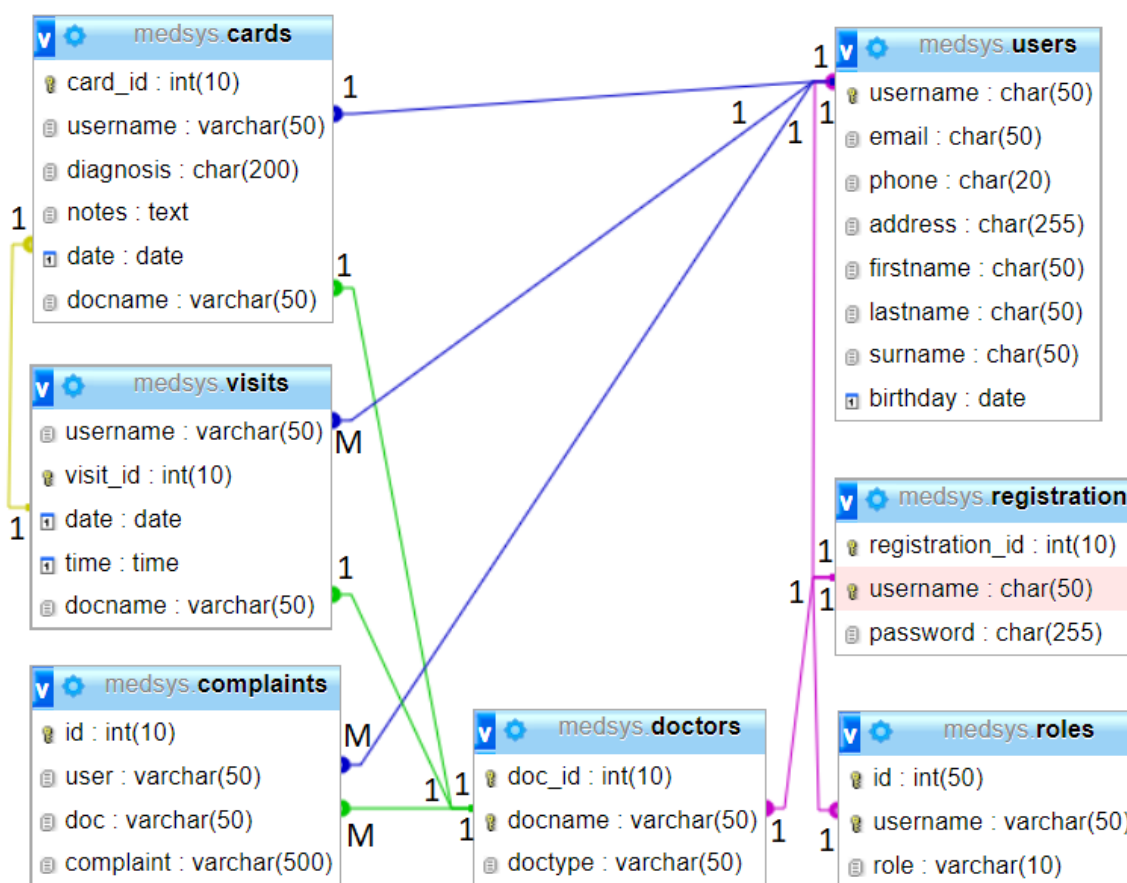


Рисунок 2.6 Логическая модель базы данных

2.2.4 Физическое проектирование

Физическая модель БД определяет способы размещения данных в среде хранения и способ доступа к этим данным, поддерживаемым на физическом уровне.

База данных «medsys» и ее таблицы были созданы SQL-запросами в PHPMyAdmin.

Создание БД было описано запросом:

```
CREATE DATABASE medsys;
```

Далее представлены коды создания таблиц в этой базе данных (registration, roles, users, doctors, visits, cards, complaints) [10]:

1) Таблица «registration»

```
CREATE TABLE IF NOT EXISTS `registration` (  
  `registration_id` int(10) NOT NULL AUTO_INCREMENT,  
  `username` char(50) CHARACTER SET cp1251 NOT NULL,  
  `password` char(255) CHARACTER SET cp1251 NOT NULL,  
  PRIMARY KEY (`registration_id`),  
  UNIQUE KEY `username` (`username`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

2) Таблица «roles»

```
CREATE TABLE IF NOT EXISTS `roles` (  
  `id` int(50) NOT NULL AUTO_INCREMENT,  
  `username` varchar(50) CHARACTER SET cp1251 NOT NULL,  
  `role` varchar(10) CHARACTER SET cp1251 NOT NULL,  
  PRIMARY KEY (`id`),  
  UNIQUE KEY `username` (`username`),  
  KEY `role` (`role`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

3) Таблица «users»

```
CREATE TABLE IF NOT EXISTS `users` (  
  `username` char(50) CHARACTER SET cp1251 NOT NULL,  
  `email` char(50) CHARACTER SET cp1251 NOT NULL,  
  `phone` char(20) CHARACTER SET cp1251 NOT NULL,  
  `address` char(255) CHARACTER SET cp1251 NOT NULL,  
  `firstname` char(50) CHARACTER SET cp1251 NOT NULL,  
  `lastname` char(50) CHARACTER SET cp1251 NOT NULL,  
  `surname` char(50) CHARACTER SET cp1251 NOT NULL,  
  `birthday` date NOT NULL,  
  PRIMARY KEY (`username`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

4) Таблица «doctors»

```
CREATE TABLE IF NOT EXISTS `doctors` (
  `doc_id` int(10) NOT NULL AUTO_INCREMENT,
  `docname` varchar(50) NOT NULL,
  `doctype` varchar(50) NOT NULL,
  PRIMARY KEY (`doc_id`),
  UNIQUE KEY `docname` (`docname`),
  UNIQUE KEY `docname_2` (`docname`)
) ENGINE=InnoDB DEFAULT CHARSET=cp1251;
```

5) Таблица «visits»

```
CREATE TABLE IF NOT EXISTS `visits` (
  `username` varchar(50) CHARACTER SET cp1251 NOT NULL,
  `visit_id` int(10) NOT NULL AUTO_INCREMENT,
  `date` date NOT NULL,
  `time` time NOT NULL,
  `docname` varchar(50) CHARACTER SET cp1251 NOT NULL,
  PRIMARY KEY (`visit_id`),
  KEY `username` (`username`),
  KEY `docname` (`docname`),
  KEY `date` (`date`),
  KEY `docname_2` (`docname`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

6) Таблица «cards»

```
CREATE TABLE IF NOT EXISTS `cards` (
  `card_id` int(10) NOT NULL AUTO_INCREMENT,
  `username` varchar(50) CHARACTER SET cp1251 NOT NULL,
  `diagnosis` char(200) CHARACTER SET cp1251 NOT NULL,
  `notes` text CHARACTER SET cp1251 NOT NULL,
  `date` date NOT NULL,
  `docname` varchar(50) CHARACTER SET cp1251 NOT NULL,
  PRIMARY KEY (`card_id`),
  KEY `username` (`username`),
  KEY `docname` (`docname`),
  KEY `date` (`date`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

7) Таблица «complaints»

```

CREATE TABLE IF NOT EXISTS `complaints` (
  `id` int(10) NOT NULL AUTO_INCREMENT,
  `user` varchar(50) NOT NULL,
  `doc` varchar(50) NOT NULL,
  `complaint` varchar(500) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `user` (`user`),
  KEY `doc` (`doc`)
) ENGINE=InnoDB DEFAULT CHARSET=cp1251;

```

После создания таблиц в базе данных и обозначения ключевых атрибутов в каждой из них необходимо также добавить ограничения внешних ключей. Это было сделано при помощи кода, приведенного далее:

```
-- Ограничения внешнего ключа таблицы `complaints`
```

```

ALTER TABLE `complaints` ADD CONSTRAINT `complaints_ibfk_1`
FOREIGN KEY (`doc`) REFERENCES `doctors` (`docname`), ADD CONSTRAINT
`complaints_ibfk_2` FOREIGN KEY (`user`) REFERENCES `users` (`username`);

```

```
-- Ограничения внешнего ключа таблицы `doctors`
```

```

ALTER TABLE `doctors` ADD CONSTRAINT `doctors_ibfk_1` FOREIGN KEY
(`docname`) REFERENCES `registration` (`username`);

```

```
-- Ограничения внешнего ключа таблицы `roles`
```

```

ALTER TABLE `roles` ADD CONSTRAINT `roles_ibfk_1` FOREIGN KEY
(`username`) REFERENCES `registration` (`username`);

```

```
-- Ограничения внешнего ключа таблицы `users`
```

```

ALTER TABLE `users` ADD CONSTRAINT `users_ibfk_1` FOREIGN KEY
(`username`) REFERENCES `registration` (`username`);

```

```
-- Ограничения внешнего ключа таблицы `visits`
```

```

ALTER TABLE `visits` ADD CONSTRAINT `visits_ibfk_2` FOREIGN KEY
(`docname`) REFERENCES `doctors` (`docname`), ADD CONSTRAINT `visits_ibfk_3`
FOREIGN KEY (`username`) REFERENCES `users` (`username`);

```

Получившаяся структура наглядно представлена на рисунках 2.7-2.13.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	<u>registration_id</u>	int(10)			Нет	Нет	AUTO_INCREMENT
2	username	char(50)	cp1251_general_ci		Нет	Нет	
3	password	char(255)	cp1251_general_ci		Нет	Нет	

Рисунок 2.7 Структура таблицы «registration»

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	<u>id</u>	int(50)			Нет	Нет	AUTO_INCREMENT
2	username	varchar(50)	cp1251_general_ci		Нет	Нет	
3	role	varchar(10)	cp1251_general_ci		Нет	Нет	

Рисунок 2.8 Структура таблицы «roles»

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	<u>username</u>	char(50)	cp1251_general_ci		Нет	Нет	
2	email	char(50)	cp1251_general_ci		Нет	Нет	
3	phone	char(20)	cp1251_general_ci		Нет	Нет	
4	address	char(255)	cp1251_general_ci		Нет	Нет	
5	firstname	char(50)	cp1251_general_ci		Нет	Нет	
6	lastname	char(50)	cp1251_general_ci		Нет	Нет	
7	surname	char(50)	cp1251_general_ci		Нет	Нет	
8	birthday	date			Нет	Нет	

Рисунок 2.9 Структура таблицы «users»

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	<u>doc_id</u>	int(10)			Нет	Нет	AUTO_INCREMENT
2	docname	varchar(50)	cp1251_general_ci		Нет	Нет	
3	doctype	varchar(50)	cp1251_general_ci		Нет	Нет	

Рисунок 2.10 Структура таблицы «doctors»

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	username	varchar(50)	cp1251_general_ci		Нет	Нет	
2	<u>visit_id</u>	int(10)			Нет	Нет	AUTO_INCREMENT
3	date	date			Нет	Нет	
4	time	time			Нет	Нет	
5	docname	varchar(50)	cp1251_general_ci		Нет	Нет	

Рисунок 2.11 Структура таблицы «visits»

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	card_id	int(10)			Нет	Нет	AUTO_INCREMENT
2	username	varchar(50)	cp1251_general_ci		Нет	Нет	
3	diagnosis	char(200)	cp1251_general_ci		Нет	Нет	
4	notes	text	cp1251_general_ci		Нет	Нет	
5	date	date			Нет	Нет	
6	docname	varchar(50)	cp1251_general_ci		Нет	Нет	

Рисунок 2.12 Структура таблицы «cards»

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	id	int(10)			Нет	Нет	AUTO_INCREMENT
2	user	varchar(50)	cp1251_general_ci		Нет	Нет	
3	doc	varchar(50)	cp1251_general_ci		Нет	Нет	
4	complaint	varchar(500)	cp1251_general_ci		Нет	Нет	

Рисунок 2.13 Структура таблицы «complaints»

2.3 Разработка веб-приложения

Разработанное в рамках настоящей ВКР приложение представляет собой множество html-страниц, хранящихся на локальном сервере. Страницы связаны между собой таким образом, чтобы при взаимодействии с интерфейсом пользователь мог переключаться между ними. При переключении с одной страницы на другую при необходимости также передаются некоторые данные.

Как известно, для написания веб-приложений не требуется никаких «особенных» программ, достаточно самого обычного текстового редактора, пусть даже «Блокнот». Для удобства восприятия кода при выполнении работы был использован редактор «Notepad++».

Для написания приложения был использован язык разметки гипертекста (HTML), каскадные таблицы стилей (CSS), языки программирования JavaScript и PHP.

Тестирование выполнялось посредством запуска приложения через веб-браузер и проверки его работоспособности вручную.

Далее будут отмечены этапы разработки, на которые хотелось бы обратить внимание.

2.3.1 Разработка Frontend-части

Основные элементы интерфейса: интерактивное многоуровневое меню, модальное окно регистрации со вкладками для переключения между регистрацией и входом в систему, формы, поля ввода различных типов, кнопки. Также на главной странице приложения присутствует слайдер. Его задумка состоит в том, чтобы иметь возможность оформить начальную страницу приложения под конкретное медицинское учреждение. Например, на слайдах можно разместить фотографии учреждения и информацию о нем.

Для создания слайдера потребовалось реализовать несколько переключателей, чтобы листать слайды, и сами слайды с фоновыми изображениями. Его код приведен в приложении 1. Слайдер написан на HTML и CSS [11]. Для корректного отображения изображений они были помещены в папку images рядом с файлами с кодом.

Интерактивное меню содержит два уровня, то есть существуют такие пункты меню, при выборе которых нужно будет совершить выбор еще раз из выпадающего списка. Меню написано на HTML с подключением CSS, код представлен в приложении 2. Реализация осуществления переходов между страницами при помощи меню относится уже к backend-части и описана в следующем пункте данной работы.

Чтобы перейти с главной страницы в соответствующий роли пользователя раздел приложения нужно выполнить вход. Если пользователь существует, тогда система проверит введенные логин и пароль, распознает роль и позволит выполнить переход. Если пользователя не существует, нужно его создать. Таким образом, нужна форма регистрации и авторизации пользователей. Она была реализована в виде модального окна с двумя вкладками. В приложении 3 представлен код модального окна регистрации и авторизации [12, 13].

Далее нужно обратить внимание на кастомизацию стиля полей и кнопок, которые будут основой форм сбора данных для обработки. В HTML поле ввода создается при помощи тега `input`. В описании поля указывается, как минимум, его имя, тип (например, `text` или `date`), значение по умолчанию, а также подключается стиль при помощи `class`. Многострочные текстовые поля создаются тегом `textarea`.

Для полей с выпадающим списком используется тег `select`. Кнопка создается аналогично, но вместо значения по умолчанию будет задаваться надпись на кнопке, а тип будет `submit`. Также при помощи тега `input` создаются кнопки типа `radio`. В приложении 4 и приложении 5 представлены примеры объявления полей и кнопок, а также подключенные к ним стили.

Для удобного представления данных разработанное веб-приложение в некоторых случаях использует таблицы. Они создаются при помощи тега `table`.

Также стандартный шрифт был заменен на `Courier`, он подгружается из Интернета. В случае отсутствия подключения к сети Интернет текст будет отображаться тем шрифтом, который указан замещающим. В данном случае это `Arial`.

2.3.2 Разработка Backend-части

Backend – это обработка данных, полученных в результате взаимодействия пользователя с интерфейсом. Эту часть программы пользователь не видит. Она работает в тесной связи с frontend-частью. Для ее реализации использовались языки программирования PHP и JavaScript.

Код разработанного веб-приложения довольно объемный, поэтому здесь описаны только основные конструкции, которые были написаны в процессе разработки. Примеры кода приведены в приложениях 6-8.

Для передачи данных между страницами использовался механизм `session` (приложение 6). Такая необходимость возникает, например, после авторизации: нужно временно хранить логин пользователя, чтобы в профиле отобразилась его информация, нужно запоминать логин пациента, чтобы при записи на прием, отправке жалобы или письма врачу не пришлось вводить его повторно, и так далее.

Проект основан на работе с базой данных, соответственно, нужно выполнять подключение к локальному серверу, где хранится база данных, а затем к самой базе данных. Во избежание некорректного чтения данных нужно также указать кодировку, в соответствии с которой они будут распознаваться [4].

Стоит отметить, что в данной работе необходимо хранение данных для авторизации пользователей в базе данных. Для обеспечения безопасности нельзя хранить пароли в открытом виде, поэтому был использован механизм шифрования MD5. Таким образом, в таблице базы данных хранятся хэши паролей, а не сами

пароли, а при авторизации введенный пользователем пароль зашифровывается и сравнивается с хранимым хэшем.

При работе с БД использовались следующие операторы:

1. SELECT. Оператор чтения, позволяет делать выборку записей. Используется, например, для вывода таблицы приемов, на которые записывался пациент.
2. UPDATE. Служит для редактирования записей в таблицах. Применяется, например, для редактирования информации профиля пользователей.
3. INSERT. Добавляет запись в таблицу. Такая необходимость появляется, например, при регистрации нового пользователя в системе: создается новый аккаунт, и информация о нем записывается в базу данных.
4. WHERE. Позволяет выделить записи по заданным критериям, чтобы совершать действия только над ними.
5. COUNT. Используется внутри SELECT-выборки. Подсчитывает количество записей в выборке.
6. CONCAT. Позволяет осуществлять конкатенацию строк в запросе. В данной работе, например, с помощью этого оператора было реализовано получение строки «Фамилия Имя Отчество» из отдельно хранящихся в таблице фамилии, имени и отчества.
7. JOIN. Позволяет работать с данными нескольких таблиц одновременно, соединяя их по определенному соответствию записей. Структура таблиц, участвующих в запросе с JOIN, не меняется.
8. AS. В ответе на запрос заменяет фактическое наименование столбца таблицы на заданное, внутри базы данных при этом никаких изменений не происходит [10].

В результате успешного запроса к БД придет ответ в формате assoc. Этот формат отображает пары «атрибут-значение». Позволяет быстро получать значение по названию атрибута.

Примеры работы с базой данных представлены в приложении 7.

Также была изучена технология отправки писем на электронную почту. Для этого необходимо задать адрес отправителя и получателя, тему письма и текст письма (приложение 8). В Localhost Denwer присутствует «sendmail-заглушка», то есть функция отправки сообщения на электронную почту доступна только в режиме

отладки, поскольку отладка – основное его назначение. Генерируемые письма сохраняются на виртуальном локальном сервере. Фрагмент кода, генерирующий письмо, приведен в приложении 8 [5].

Выводы

Итак, в практической части данной выпускной квалификационной работы были описаны возможности разработанного веб-приложения, этапы проектирования базы данных для него, обозначены назначение и функции БД, построена инфологическая и концептуальная модель, выполнена нормализация, описаны схемы таблиц на SQL и наглядно представлены их структуры, затем был описан процесс разработки CRM-системы для организации консультации в медучреждениях с использованием HTML, CSS, PHP, JavaScript.

Разработанное веб-приложение предоставляет пользователям определенные возможности в зависимости от их роли, в основе которых лежит работа с базой данных. Администратору – редактировать информацию своего профиля, добавлять новых пользователей системы под любую роль, просматривать список жалоб. Пациенту – редактировать информацию профиля, просматривать таблицу последних посещений, записываться на прием, просматривать медкарту, отправить письмо врачу на электронную почту, оставить жалобу. Врачу – редактировать профиль, медицинские карты по результатам приема, просматривать таблицу записей на прием.

Раздел 3. Пример работы приложения

3.1 Запуск приложения. Главная страница

При открытии веб-приложения с сервера перед глазами пользователя предстает страница `index.php`, являющаяся стартовой. На ней присутствует полноэкранный слайдер и возможность регистрации или авторизации в системе. Авторизоваться может пользователь с любой ролью. Форма регистрации позволяет регистрироваться только пациентам. Новых администраторов и врачей может добавить только действующий администратор.

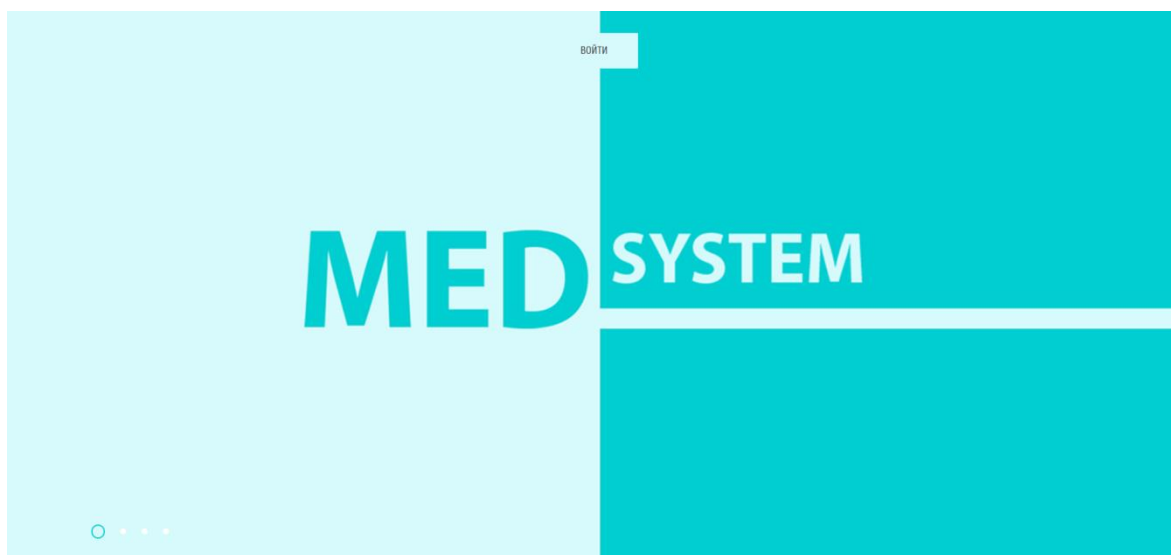


Рисунок 3.1 Стартовая страница

Рисунок 3.2 Модальное окно, вкладка авторизации

Требуется авторизация X

Вход
Регистрация

Имя пользователя

E-mail

Пароль

Подтвердите пароль

OK
Сброс

Необходимо заполнить все поля

Рисунок 3.3 Модальное окно, вкладка регистрации

3.2 Возможности личного кабинета пользователя с ролью «администратор»

ПРОФИЛЬ
ДОБАВИТЬ ...
ЖАЛОБЫ
ВЫЙТИ

Администратора
Врача
Пациента

Рисунок 3.4 Меню личного кабинета администратора

Пункт меню «Профиль» позволяет перейти к интерфейсу просмотра и редактирования личной информации.

Логин:

Фамилия:

Имя:

Отчество:

E-mail:

Номер телефона:

Адрес:

Редактировать

Рисунок 3.5 Профиль администратора

Администратор имеет возможность добавлять новых пользователей, причем для разных ролей разный набор необходимых к заполнению полей. При желании более подробную информацию пользователь сможет указать самостоятельно в разделе «Профиль» своего личного кабинета.

Form for adding a new administrator. It contains two input fields: 'Логин:' (Login) and 'Пароль:' (Password). Below the fields is a blue button labeled 'Добавить' (Add).

Рисунок 3.6 Интерфейс добавления нового администратора

Form for adding a new administrator. It contains seven input fields: 'Логин:' (Login), 'Пароль:' (Password), 'E-mail:', 'Фамилия:' (Surname), 'Имя:' (Name), 'Отчество:' (Patronymic), and 'Должность:' (Position). Below the fields is a blue button labeled 'Добавить' (Add).

Рисунок 3.7 Интерфейс добавления нового врача

Form for adding a new doctor. It contains eight input fields: 'Логин:' (Login), 'Пароль:' (Password), 'Фамилия:' (Surname), 'Имя:' (Name), 'Отчество:' (Patronymic), 'Дата рождения:' (Date of Birth), 'Телефон:' (Phone), and 'Адрес:' (Address). Below the fields is a blue button labeled 'Добавить' (Add).

Рисунок 3.8 Интерфейс добавления нового пациента

Для просмотра жалоб, оставленных пациентами, администратор может выполнить переход в раздел «Жалобы», воспользовавшись соответствующим пунктом меню.

Врач	Пациент	Жалоба
Фролов Василий Львович	Николаева Анна Васильевна	Врач хлопнул дверью!
Фролов Василий Львович	Николаева Анна Васильевна	В кабинете некрасиво постриженные кусты! Никакого эстетического наслаждения! Я негодную!

Рисунок 3.9 Просмотр жалоб пациентов

3.3 Возможности личного кабинета пользователя с ролью «врач»

Меню пользователя с ролью «врач» содержит три основных пункта: «Профиль», «Посещения», «Выйти». Пункт «Посещения» содержит два подпункта: «Таблица приемов» и «Карты». Меню представлено на рисунке 3.10.

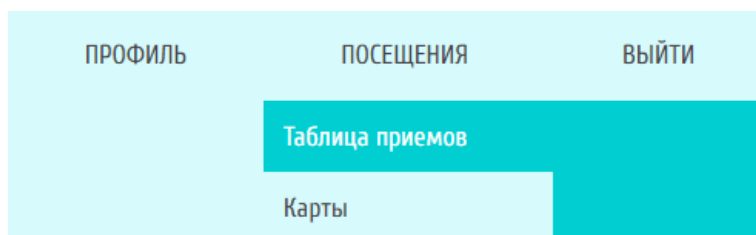


Рисунок 3.10 Меню личного кабинета врача

Как и в случае с пользователем-администратором, раздел «Профиль» предоставляет возможность изменить личную информацию, однако у врача будет гораздо больше предзаполненной информации, чем у администратора, поскольку при регистрации администратором нового врача необходимо подавать больше информации, чем при регистрации нового администратора.


Логин:	<input type="text" value="Dr.Batya"/>
Фамилия:	<input type="text" value="Фролов"/>
Имя:	<input type="text" value="Василий"/>
Отчество:	<input type="text" value="Львович"/>
Дата рождения:	<input type="text" value="17.04.2022"/> 
E-mail:	<input type="text" value="batya@mail.ru"/>
Номер телефона:	<input type="text" value="09991112233"/>
Адрес:	<input type="text" value="г.Санкт-Петербург, ул.Наличная, д.21, кв.56"/>

Рисунок 3.11 Раздел «Профиль» личного кабинета врача

При переходе в раздел «Таблица приемов» система запросит дату. Это дата, приемы за которую нужно вывести. Причем выводиться будут только приемы, относящиеся к врачу, который делает запрос.


Дата:	<input type="text" value="дд.мм.гггг"/> 
-------	---

Рисунок 3.12 Ввод даты для отбора таблицы приемов по дате

Дата	Время	Фамилия пациента	Имя пациента	Отчество пациента
2022-04-14	09:00:00	Николаева	Анна	Васильевна
2022-04-14	08:00:00	Николаева	Анна	Васильевна
2022-04-14	00:00:00	Николаева	Анна	Васильевна
2022-04-14	00:00:00	Николаева	Анна	Васильевна
2022-04-14	00:00:00	Николаева	Анна	Васильевна
2022-04-14	09:30:00	Николаева	Анна	Васильевна

Рисунок 3.13 Пример результата вывода списка приемов по дате

В день приема врач имеет возможность отредактировать медицинскую карту. В этом ему помогает интерфейс, доступный при переходе по пункту «Карты» главного меню. Сначала необходимо выбрать дату и время приема, затем появится интерфейс просмотра или редактирования медкарты в зависимости от того, выполняется переход в день приема или нет.

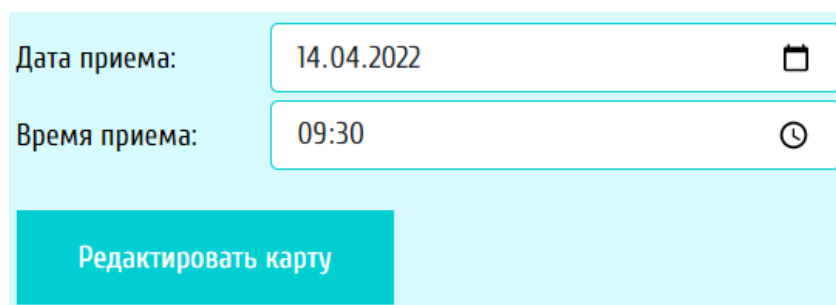


Рисунок 3.14 Указание времени и даты приема для доступа к медкарте

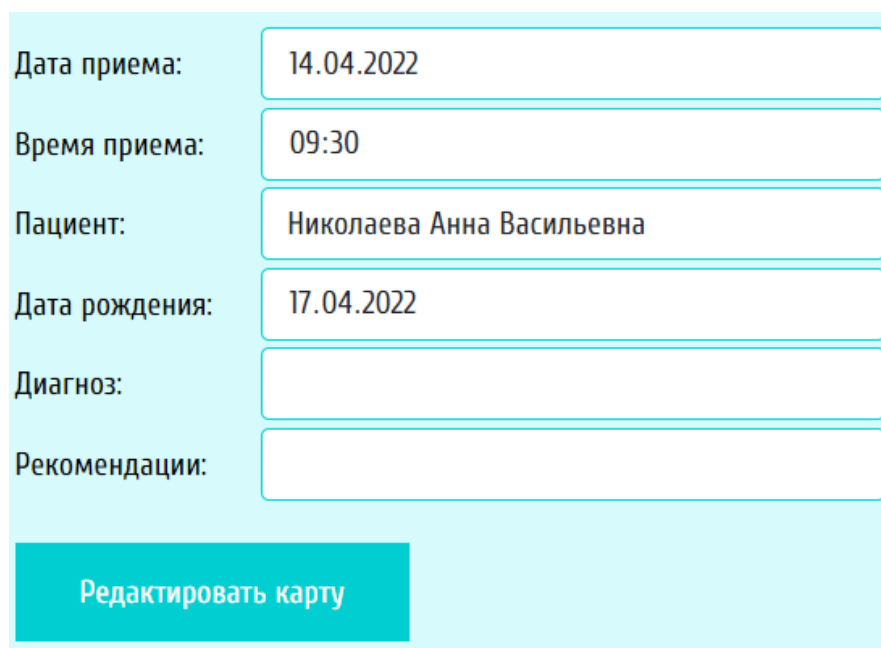


Рисунок 3.15 Интерфейс редактирования медицинской карты пациента

3.4 Возможности личного кабинета пользователя с ролью «пациент»

У пациента самый широкий спектр возможностей в разработанной системе. Меню пользователя с ролью «пациент» представлено на рисунке 3.16.



Рисунок 3.16 Меню личного кабинета пациента

Пункт меню «Профиль» позволяет открыть интерфейс редактирования личной информации – ровно так же, как в личном кабинете врача или администратора. Если пациент был создан администратором, вся информация о нем будет предзаполнена. Иначе, чтобы получить услугу в медучреждении, необходимо будет заполнить всю информацию самостоятельно.


Логин:	<input type="text" value="customer"/>
Фамилия:	<input type="text" value="Николаева"/>
Имя:	<input type="text" value="Анна"/>
Отчество:	<input type="text" value="Васильевна"/>
Дата рождения:	<input type="text" value="17.04.2022"/> 
E-mail:	<input type="text" value="nikav@mail.ru"/>
Номер телефона:	<input type="text" value="12223334455"/>
Адрес:	<input type="text" value="г.Санкт-Петербург, ул.Наличная, д.36, кв.4"/>
<input type="button" value="Редактировать"/>	

Рисунок 3.17 Раздел «Профиль» пациента

При выполнении перехода «Посещения» -> «Последние визиты» появится перечень последних записей на прием текущего пациента.

Врач	Фамилия врача	Имя врача	Отчество врача	Дата	Время
терапевт	Фролов	Василий	Львович	2022-05-20	00:00:00
терапевт	Господин	Ибрагим	Андреевич	2022-05-20	12:00:00
терапевт	Фролов	Василий	Львович	2022-05-18	00:00:00
терапевт	Фролов	Василий	Львович	2022-05-18	00:00:00
терапевт	Фролов	Василий	Львович	2022-05-18	00:00:00
терапевт	Фролов	Василий	Львович	2022-04-14	00:00:00
терапевт	Фролов	Василий	Львович	2022-04-14	08:00:00
терапевт	Фролов	Василий	Львович	2022-04-14	09:00:00
терапевт	Фролов	Василий	Львович	2022-04-14	09:30:00
терапевт	Фролов	Василий	Львович	2022-04-14	00:00:00


Рисунок 3.18 Вывод перечня записей пациента на прием ко врачу

Для записи на прием в соответствующем разделе необходимо задать должность врача, например, «терапевт», и дату, на которую желательна запись.

Должность
врача:

терапевт

Дата:

20.05.2022 

Далее

Рисунок 3.19 Запись на прием, выбор должности врача и даты

Следующим этапом система выберет врачей по указанной должности из базы данных и предложит выбрать одного из них при помощи выпадающего списка.

Должность врача:	терапевт
Дата:	20.05.2022
ФИО врача:	<div>▼</div> <div> Фролов Василий Львович Господин Ибрагим Андреевич </div>
<div>Далее</div>	

Рисунок 3.20 Запись на прием, выбор врача

После выбора врача откроется интерфейс выбора времени приема. Время, которое уже занято, будет заблокировано для выбора. На рисунке 3.21 видно, что недоступно время «12:00».

Должность врача:	терапевт
ФИО врача:	Господин Ибрагим Андреевич
Дата:	20.05.2022
Время:	<div> <input type="radio"/> 8:00 <input type="radio"/> 8:30 <input type="radio"/> 9:00 <input type="radio"/> 9:30 <input checked="" type="radio"/> 10:00 <input type="radio"/> 10:30 <input type="radio"/> 11:00 <input type="radio"/> 11:30 <input type="radio"/> 12:00 <input type="radio"/> 12:30 <input type="radio"/> 13:00 <input type="radio"/> 13:30 <input type="radio"/> 14:00 <input type="radio"/> 14:30 <input type="radio"/> 15:00 <input type="radio"/> 15:30 </div>
<div>Записаться</div>	

Рисунок 3.21 Запись на прием, выбор времени

Пациент имеет возможность посмотреть результаты своего похода на прием ко врачу в разделе «Медкарта». Сначала необходимо будет указать должность врача и дату посещения (рисунок 3.22), затем по нажатию на кнопку выведется медкарта (рисунок 3.23).


Должность врача:	терапевт
Дата:	17.12.2021 
<div>Посмотреть карту</div>	

Рисунок 3.22 Выбор должности врача и даты приема, за которую создана запись в медкарте

Врач	Фамилия врача	Имя врача	Отчество врача	Дата	Диагноз	Рекомендации
терапевт	Фролов	Василий	Львович	2021-12-17	опа	вот так

Рисунок 3.23 Просмотр результата приема

Предусмотрен функционал связи со врачом посредством отправки письма на его электронную почту прямо из веб-приложения. Для этого сначала надо выбрать должность врача (рисунок 3.24), затем – ФИО врача из сформированного в результате запроса к БД выпадающего списка. По завершении написания текста письма сообщение можно отправить (рисунок 3.25).

Должность врача:

Далее

Рисунок 3.24 Поле для указания должности врача

Логин:

Должность врача:

ФИО врача:

Текст письма:

Отправить

Рисунок 3.25 Форма отправка электронного письма врачу

В результате сгенерируется письмо вида, показанного на рисунке 3.26.

```
X-Sendmail-Cmdline: sendmail.pl -t -i
To: monster@mail.ru
Subject: письмо от пациента
X-PHP-Originating-Script: 0:sendletter.php
Content-type:text/plain; charset = utf-8
From:nikav@mail.ru

Текст письма ...
```

Рисунок 3.26 Сгенерированное письмо

Интерфейс отправки жалобы такой же, как и интерфейс отправки письма врачу (рисунок 3.27, 3.28). Но в случае с жалобой выполняется запись в базу данных, после чего отзыв пациента может увидеть администратор.

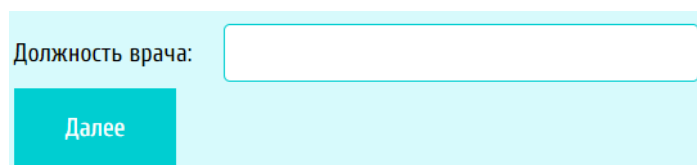


Рисунок 3.27 Поле для указания должности врача перед отправкой жалобы

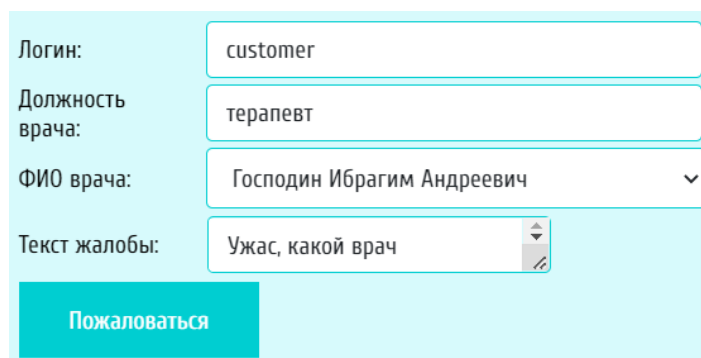


Рисунок 3.28 Интерфейс отправки жалобы

Выводы

В данном разделе продемонстрированы примеры работы всех частей разработанного приложения, описаны все его возможности и особенности, а также представлена инструкция по работе с приложением.

Разработано простое и удобное приложение с полным функционалом, выгодно отличающееся от существующих на рынке аналогов.

Раздел 4. Экономическая часть

Основная цель экономического раздела работы объясняет целесообразность реализации предложенного веб-приложения путем оценки его эффективности после внедрения в организацию. Для того, чтобы оценить эффективность, необходимо провести оценку инвестиционных расходов, определить эксплуатационные расходы, определить эффект при реализации мероприятия.

Конечной целью выпускной квалификационной работы является разработка CRM-системы для организации консультаций в медучреждениях, основная идея которой – расширение возможностей коммуникации пациента с медучреждением.

4.1 Оценка инвестиционных расходов

Проекты, связанные с внедрением современных информационных систем и технологий, требуют материальных и трудовых затрат. При проектировании мероприятий в сфере автоматизированных систем обработки информации и управления инвестиционные затраты определяются капитальными вложениями – инвестициями, направленными на воспроизводство основных средств.

Капитальные вложения вычисляются по формуле [6]:

$$K_p = Z_{об} + Z_{пп} + Z_{мат} + Z_{зп} + Z_{сн} + Z_{пр}, \quad (4.1)$$

где

$Z_{об}$ – затраты, связанные с приобретением оборудования или эксплуатацией техники, руб.;

$Z_{пп}$ – затраты на специальные программные продукты, которые необходимы для разработки проекта, руб.;

$Z_{мат}$ – затраты на хозяйственно-операционные нужды, руб.;

$Z_{зп}$ – общий фонд оплаты труда разработчиков, руб.;

$Z_{сн}$ – затраты на выплату страховых взносов, руб.;

$Z_{пр}$ – прочие затраты, руб.

Для расчета требуется выделить основные этапы проекта, подробно представить список работ на каждом из этапов, учитывая последовательность выполнения (таблица 4.1).

Таблица 4.1

Этап	Проводимые работы	Продолжительность, ч.
1. Постановка задачи	1. Изучение предметной области; 2. Анализ рынка.	20
2. Проектирование	1. Выбор инструментальных средств разработки; 2. Разработка структуры информационной системы; 3. Разработка дизайна системы.	70
3. Программирование	1. Создание базы данных; 2. Создание веб-приложения.	200
4. Тестирование	1. Проверка работоспособности системы и выявление ошибок; 2. Устранение выявленных ошибок.	30
Итого:	320	

Закупка дополнительного оборудования и его эксплуатация не производилась в рамках данной ВКР, следовательно затраты приравниваются к нулю: $Z_{об} = 0$.

Программные средства, использованные в ходе разработки, являются бесплатными и общедоступными, значит затраты на ПО также приравниваются к нулю: $Z_{пп} = 0$.

Далее нужно рассчитать материальные затраты по следующей формуле [6]:

$$Z_{\text{мат}} = Z_{\text{с}} + Z_{\text{м}} + Z_{\text{эл}} + Z_{\text{тр}} + Z_{\text{пр.мат}}, \quad (4.2)$$

где

$Z_{\text{с}}$ – затраты на сырье, руб.;

$Z_{\text{м}}$ – затраты на материалы, руб.;

$Z_{\text{эл}}$ – затраты на электроэнергию, руб.;

$Z_{\text{тр}}$ – затраты на транспорт, руб.;

$Z_{\text{пр.мат}}$ – прочие материальные затраты, руб.

Разработка проводилась на персональном компьютере, мощность блока питания которого составляет 120 Вт, тогда:

$$Z_{\text{эл}} = \frac{120 * k}{1000} * C * T, \quad (4.3)$$

где

k – коэффициент полезной работы;

C – стоимость киловатт-часа, руб.;

T – время использования ПК, ч.

При стоимости 1 киловатт-часа 4,98 руб.:

$$Z_{\text{эл}} = \frac{120 * 1,2}{1000} * 4,98 * 320 = 229,48 \text{ (руб.)}$$

К прочим материальным затратам относится плата за доступ к сети Интернет. Тарифный план составляет 600 руб./мес., значит $Z_{\text{пр.мат}} = 1200$ руб.

$$Z_{\text{мат}} = Z_c + Z_m + Z_{\text{эл}} + Z_{\text{тр}} + Z_{\text{пр}} = 0 + 0 + 229,48 + 0 + 1200 = 1429,48 \text{ (руб.)}$$

Далее необходимо рассчитать затраты на оплату труда разработчику проекта.

При расчете заработной платы будет использоваться повременная форма оплаты труда:

$$Z_{\text{вр}} = T * C, \quad (4.4)$$

где

T – фактически отработанное время работником, ч.;

C – часовая тарифная ставка работника, руб./ч.

Затраты на оплату труда равны сумме заработных плат всех работников проекта (таблица 4.2).

Таблица 4.2

Категория работника	Трудоемкость разработки, чел./ч.	Часовая ставка, руб./ч.	Сумма, руб.	Сумма с учетом страховых взносов, руб.
Разработчик	320	450	144000	187200

Таким образом, затраты на оплату труда $Z_{зп} = 144000$ руб.

На фонд заработной платы производится начисление страховых взносов на обязательное социальное страхование – это обязательные отчисления по установленным законодательством нормам (обязательное пенсионное страхование, страхование на случай временной нетрудоспособности и материнства, обязательное медицинское страхование):

$$Z_{сн} = Z_{зп} * K_{сн} \quad (4.5)$$

Тариф на страховые взносы в 2022 году составляет 30% от общего фонда оплаты труда, данная ставка регулируется государством (НК РФ).

Таким образом,

$$Z_{сн} = 144000 * 0,3 = 43200 \text{ (руб.)}.$$

Прочие затраты $Z_{пр}$ примем за 0.

В итоге капитальные затраты, необходимые для реализации проекта, равны:

$$K_p = 0 + 0 + 1429,48 + 144000 + 43200 + 0 = 188629,48 \text{ (руб.)}$$

4.2 Оценка эксплуатационных расходов

Эксплуатационные расходы – это годовые текущие издержки, которые связаны с эксплуатацией внедренного проекта. Они определяются по формуле [6]:

$$Z_{экспл} = Z_{мат} + Z_{зп} + Z_{сн} + A_m + Z_{пр}, \quad (4.6)$$

где

$Z_{мат}$ – материальные затраты, руб.;

$Z_{зп}$ – заработная плата, руб.;

$Z_{сн}$ – страховые взносы на обязательное социальное страхование, руб.;

A_m – амортизационные отчисления, руб.;

$Z_{пр}$ – прочие расходы, руб.

При расчёте материальных затрат вычисления выполняются по формуле 4.2.

Затраты на сырье, материалы и транспорт отсутствуют: $Z_c = 0$ руб., $Z_m = 0$ руб. и $Z_{тр} = 0$ руб.

Расчет затрат на электроэнергию осуществляется из расчета, что в 2022 году 247 рабочих дней, рабочий день длится 8 часов, а стоимость киловатт-часа составляет 4,98 рубля. Тогда согласно формуле 4.3 получим:

$$З_{эл} = \frac{120 * 1,2}{1000} * 4,98 * 247 * 8 = 1417,03 \text{ (руб.)}$$

К прочим расходам относится плата за доступ к Интернету. Тарифный план составляет 600 руб./мес. Тогда годовые затраты за доступ к Интернету:

$$З_{пр.мат.} = 600 * 12 = 7200 \text{ (руб.)}$$

Таким образом, материальные затраты составляют

$$З_{мат} = З_c + З_m + З_{эл} + З_{тр} + З_{пр} = 0 + 0 + 1417,03 + 0 + 7200 = 8617,03 \text{ (руб.)}$$

При расчете оплаты труда участвующих в эксплуатации проекта работников будет использоваться повременная форма оплаты труда. Для эксплуатации проекта будет выделен один сотрудник – администратор системы, работающий 1 час в день по тарифной ставке 350 руб./ч.

Так, затраты на оплату труда составляют

$$З_{зп} = 247 * 350 * 1 = 86450 \text{ (руб.)}$$

Страховые взносы на обязательное социальное страхование рассчитываются по формуле 4.5:

$$З_{сн} = 86450 * 0,3 = 25935 \text{ (руб.)}$$

Основные средства – это средства труда, которые участвуют в производственном процессе, сохраняя при этом свою натуральную форму, и постепенно переносят свою стоимость на себестоимость продукции в виде амортизационных отчислений.

При эксплуатации разработанной системы специальное оборудование не потребуется – достаточно уже имеющихся на предприятии персональных компьютеров работников стандартной комплектации с заранее установленной операционной системой. Значит, $A_m = 0$ (руб.).

Прочие расходы будут составлять $З_{пр} = 5000$ (руб.) – непредвиденные расходы.

Таким образом, эксплуатационные затраты за 1 год будут составлять:

$$З_{\text{экспл}} = 8617,03 + 86450 + 25935 + 0 + 5000 = 126002,03 \text{ (руб.)}$$

4.3 Оценка эффективности проекта

Годовой экономический эффект определяется экономией или прибылью, получаемой при реализации мероприятия, за вычетом эксплуатационных расходов:

$$\mathcal{E}_{\text{год}} = \mathcal{E}_{\text{экон}} + \Pi - \mathcal{E}_{\text{экспл}} \quad (4.7)$$

где

$\mathcal{E}_{\text{экон}}$ – экономия при реализации мероприятия, руб.;

Π – чистая прибыль, получаемая при реализации мероприятия, руб.;

$\mathcal{E}_{\text{экспл}}$ – эксплуатационные затраты, руб.

По результатам тестирования, созданная CRM-система для организации консультаций в медучреждениях позволяет увеличить скорость работы медучреждения в 2-3 раза. На основе анализа стоимости работ по организации консультаций в медучреждениях можно получить ориентировочную цену одной лицензии программы в размере 50000 рублей (на одно медучреждение).

Тогда планируемая выручка от продаж при данных условиях составляет $50000 * N$, где N – количество рабочих мест медицинского учреждения, на которых будет разворачиваться система. Учитывая, что установка системы рассчитана на одно рабочее место и время разработки и развертки составляет 3 месяца, планируемая выручка будет $85000 * 3 = 255000$ (руб.).

При расчете чистой прибыли следует учитывать налог на прибыль:

$$H_{\Pi} = \Pi_{\Pi} * K_{\text{нп}} \quad (4.8)$$

где

Π_{Π} – прибыль от продаж, руб.;

$K_{\text{нп}}$ – налоговая ставка (стандартно 20%).

Таким образом, $H_{\Pi} = 255000 * 0,2 = 51000$ (руб.)

В итоге чистая прибыль составит: $\Pi = 255000 - 51000 = 204000$ (руб.)

Годовой экономический эффект:

$$\mathcal{E}_{\text{год}} = 204000 - 126002,03 = 77997,97 \text{ (руб.)}$$

Показатель абсолютной эффективности капиталовложений отражает уровень прибыли, получаемый за один год с вложенного капитала:

$$\mathcal{E}_{\text{эф}} = \frac{\Pi}{K} \quad (4.9)$$

где

Π – прибыль, получаемая при реализации мероприятия (чистая), руб.;

K – капитальные вложения в проект, руб.

$$\text{Следовательно, } \mathcal{E}_{\text{эф}} = \frac{204000}{188629,48} = 1,08$$

Срок окупаемости капитальных вложений может быть рассчитан по формуле:

$$T_{\text{ок}} = \frac{1}{\mathcal{E}_{\text{эф}}} \quad (4.10)$$

$$T_{\text{ок}} = \frac{1}{1,08} = 0,925$$

Из расчетов, приведенных выше, следует вывод, что все затраты, совершенные в процессе разработки и эксплуатации, окупятся за первые 11 месяцев работы.

Выводы

В данном разделе были определены капитальные вложения и инвестиционные расходы для разработки представленного проекта, была рассчитана сумма годовых эксплуатационных расходов. Кроме того, были рассчитаны следующие показатели: себестоимость разработки, экономическая эффективность и срок окупаемости проекта. На основании полученных результатов был сделан вывод об эффективности разрабатываемой системы и приведены расчеты, подтверждающие окупаемость разрабатываемого проекта в течение первых одиннадцати месяцев его использования в медицинском учреждении.

Заключение

В результате выполнения данной выпускной квалификационной работы была создана CRM-система для организации консультаций в медучреждениях согласно поставленной задаче при помощи выбранных средств разработки и с учетом выводов, полученных в результате анализа аналогичных программных продуктов, уже существующих на рынке.

В ходе разработки были решены следующие задачи:

- 1) анализ предметной области;
- 2) обоснование необходимости создания CRM-системы для организации консультаций в медучреждениях;
- 3) формулировка требований к проектируемой системе;
- 4) выбор технологий и инструментальных средств разработки;
- 5) выбор модели данных;
- 6) проектирование базы данных для CRM-системы;
- 7) проектирование внешней модели;
- 8) оценка экономической эффективности проекта.

В аналитической части были поставлены цели и задачи, сформулировано техническое задание, исследована актуальность разработки проекта, произведен анализ существующих аналогов. В результате исследований было принято решение о необходимости разработки CRM-системы для организации консультаций в медучреждениях.

В практической части была спроектирована реляционная база данных и создана система в виде веб-приложения. Результатом проектирования БД является представление инфологической, концептуальной и физической моделей. В результате разработки было получено рабочее приложение, позволяющее удобно организовать коммуникацию пациента с медучреждением.

В третьей части работы даны указания по эксплуатации приложения и приведен пример работы всех его функций.

В экономической части были проведены расчеты затрат на инвестиционные расходы, эксплуатационные расходы и рассчитана экономическая эффективность разрабатываемого проекта. Результатом оценки эффективности стало утверждение

экономической эффективности проекта и определение срока окупаемости 11 месяцев.

Список использованной литературы

- 1) Копейкин М.В., Спиридонов В.В., Шумова Е.О. Базы данных: учебное пособие: в 2 кн. Кн. 1. – СПб.: Изд-во СЗТУ, 2010. – 247 с.
- 2) Копейкин М.В., Спиридонов В.В., Шумова Е.О. Базы данных: учебное пособие: в 2 кн. Кн. 2. – СПб.: Изд-во СЗТУ, 2010. – 219 с.
- 3) Копейкин М.В., Спиридонов В.В., Шумова Е.О., Базы данных: учебно-методический комплекс – СПб.: Изд-во СЗТУ, 2012. – 175 с.
- 4) Ташков П.А., Веб-мастеринг на 100 %: HTML, CSS, JavaScript, PHP, CMS, AJAX, раскрутка. — СПб.: Питер, 2010. — 512 с.
- 5) Разработка электронного магазина на PHP и MySQL / Пинягина О.В. – Казань: Казанский государственный университет, 2010. – 108 с.
- 6) ЭКОНОМИКА: Методические указания по выполнению экономического раздела выпускной квалификационной работы. Санкт-Петербургский горный университет / Сост.: Л.А. Николайчук. СПб, 2020. 20 с.
- 7) Pmt. Основные проекты. МИС МЕДИАЛОГ [Электронный ресурс]. Режим доступа: <https://medialog.ru/projects/> (дата обращения: 21.05.2022)
- 8) Первый Бит. Автоматизация медицинских учреждений [Электронный ресурс]. Режим доступа: <https://spb.1cbit.ru/1s-otrasli/avtomatizaciya-medicinskix-uchrezhdenij/> (дата обращения: 21.05.2022)
- 9) ТрастМед:МИС SaaS [Электронный ресурс]. Режим доступа: <https://softrust.ru/products/trustmed-mis/> (дата обращения: 21.05.2022)
- 10) MySQL - справочное руководство [Электронный ресурс]. Режим доступа: <http://phpclub.ru/mysql/doc/index.html> (дата обращения: 21.05.2022)
- 11) Создаем полноэкранный слайдер [Электронный ресурс]. Режим доступа: <https://mnogoblog.ru/polnoekrannyj-slaider> (дата обращения: 14.03.2022)
- 12) How TO - Tabs [Электронный ресурс]. Режим доступа: https://www.w3schools.com/howto/howto_js_tabs.asp (дата обращения: 14.03.2022)
- 13) Как сделать модальное окно на сайте [Электронный ресурс]. Режим доступа: <https://smartlanding.biz/kak-sdelat-modalnoe-okno-na-sajte.html> (дата обращения: 14.03.2022)

Код слайдера (HTML)

```
<div class="css-slider-wrapper">
  <input type="radio" name="slider" class="slide-radio1" checked
id="slider_1">
  <input type="radio" name="slider" class="slide-radio2"
id="slider_2">
  <input type="radio" name="slider" class="slide-radio3"
id="slider_3">
  <input type="radio" name="slider" class="slide-radio4"
id="slider_4">
  <div class="slider-pagination">
    <label for="slider_1" class="page1"></label>
    <label for="slider_2" class="page2"></label>
    <label for="slider_3" class="page3"></label>
    <label for="slider_4" class="page4"></label>
  </div>
  <div class="slider slide-1">
    
  </div>
  <div class="slider slide-2">
    
  </div>
  <div class="slider slide-3">
    
  </div>
  <div class="slider slide-4">
    
  </div>
</div>
```

Код слайдера (CSS)

```
.css-slider-wrapper {
  display: block;
  background: #FFF;
  overflow: hidden;
  position: absolute;
  left: 0;
  right: 0;
  top: 0;
  bottom: 0;
}
/* Slider */
.slider {
  width: 100%;
  height: 100%;
  position: absolute;
  left: 0;
  top: 0;
  opacity: 1;
  z-index: 0;
  display: flex;
  flex-direction: row;
  flex-wrap: wrap;
  align-items: center;
  justify-content: center;
  align-content: center;
  -webkit-transition: -webkit-transform 1600ms;
  transition: -webkit-transform 1600ms, transform 1600ms;
```



```

        -webkit-transform: scale(1);
        transform: scale(1);
    }

    /* Slides Background Color */
    .slide-1 {
        background: #fbad99;
        left: 0;
    }
    .slide-2 {
        background: #a9785c;
        left: 100%
    }
    .slide-3 {
        background: #9ea6b3;
        left: 200%
    }
    .slide-4 {
        background: #b1a494;
        left: 300%;
    }
    .slider {
        display: flex;
        justify-content: flex-start;
    }
    /* Slider Inner Slide Effect */
    .slider > img {
        position: absolute;
        right: 10%;
        bottom: 0;
        height: 100%;
        opacity: 0;
        -webkit-transform: translateX(500px);
        transform: translateX(500px);
    }

    .slide-1 > img {
        right: 0;
    }

    .navigation .login-btn:focus {
        outline: none;
    }

    /* Animations */
    .slider > img {
        -webkit-transition: opacity 800ms, -webkit-transform 800ms;
        transition: transform 800ms, opacity 800ms;
        -webkit-transition-delay: 1.2s; /* Safari */
        transition-delay: 1.2s;
    }

    /* Slider Pagger */
    .slider-pagination {
        position: absolute;
        bottom: 30px;
        width: 575px;
        left: 100px;
        z-index: 1000;
        display: flex;
        align-items: center;
    }

```

```

/* Slider Pager Event */
.slide-radio1:checked ~ .slider-pagination .page1,
.slide-radio2:checked ~ .slider-pagination .page2,
.slide-radio3:checked ~ .slider-pagination .page3,
.slide-radio4:checked ~ .slider-pagination .page4 {
    width: 14px;
    height: 14px;
    border: 2px solid #00ced1;
    background: transparent;
}

/* Slider Slide Effect */
.slide-radio1:checked ~ .slider {
    -webkit-transform: translateX(0%);
    transform: translateX(0%);
}
.slide-radio2:checked ~ .slider {
    -webkit-transform: translateX(-100%);
    transform: translateX(-100%);
}
.slide-radio3:checked ~ .slider {
    -webkit-transform: translateX(-200%);
    transform: translateX(-200%);
}
.slide-radio4:checked ~ .slider {
    -webkit-transform: translateX(-300%);
    transform: translateX(-300%);
}

.slide-radio1:checked ~ .slide-1 h2,
.slide-radio2:checked ~ .slide-2 h2,
.slide-radio3:checked ~ .slide-3 h2,
.slide-radio4:checked ~ .slide-4 h2,
.slide-radio1:checked ~ .slide-1 h4,
.slide-radio2:checked ~ .slide-2 h4,
.slide-radio3:checked ~ .slide-3 h4,
.slide-radio4:checked ~ .slide-4 h4,
.slide-radio1:checked ~ .slide-1 &gt; img,
.slide-radio2:checked ~ .slide-2 &gt; img,
.slide-radio3:checked ~ .slide-3 &gt; img,
.slide-radio4:checked ~ .slide-4 &gt; img {
    -webkit-transform: translateX(0);
    transform: translateX(0);
    opacity: 1
}

```

Пример кода меню (HTML)

```
<nav class="dws-menu">
  <ul>
    <li><a href="userlk.php">Профиль</a></li>
    <li><a href="#">Посещения . . .</a>
    <ul>
      <li><a href="showvisits.php">Последние визиты</a></li>
      <li><a href="checkvisit.php">Запись на прием</a></li>
    </ul>
    </li>
    <li><a href="showcard.php">Медкарта</a></li>
    <li><a href="letter.php">Письмо врачу</a></li>
    <li><a href="complain.php">Пожаловаться</a></li>
    <li><a href="index.php">Выйти</a></li>
  </ul>
</nav>
```

Код меню (CSS)

```
.dws-menu *{
  margin: 0;
  padding: 0;
}
.dws-menu ul,
.dws-menu ol{
  list-style: none;
}
.dws-menu > ul{
  display: flex;
  justify-content: center;
}
.dws-menu > ul li{
  position: relative;
}
.dws-menu > ul li > a i.fa{
  position: absolute;
  top: 15px;
  left: 12px;
  font-size: 18px;
}
.dws-menu > ul li a{
  display: block;
  background: #d7fafc;
  padding: 15px 40px 15px 40px;
  font-size: 14px;
  color: #454547;
  text-decoration: none;
  text-transform: uppercase;
  transition: all 0.3s ease;
}
.dws-menu li a:hover{
  background: #00ced1;
  color: #ffffff;
  border: #d7fafc;
  transition: all 0.3s ease;
}
/*sub menu*/
```

```
.dws-menu li ul{
  position: absolute;
  min-width: 150px;
  display: none;
}
.dws-menu li > ul li a{
  padding: 10px;
  text-transform: none;
  background: #d7fafc;
}
.dws-menu li > ul li ul{
  position: absolute;
  right: -150px;
  top: 0;
}
.dws-menu li:hover > ul{
  display: block;
}
```

Код модального окна с вкладками (HTML)

```

<div id="openModal" class="modal">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h3 class="modal-title">Требуется
авторизация</h3>
        <a href="#close" title="Close"
class="close">x</a>
      </div>
      <div class="modal-body">
        <div class="tab">
          <button class="tablinks"
onclick="openEvent(event, 'SignIn')"><h3>Вход</h3></button>
          <button class="tablinks"
onclick="openEvent(event, 'SignUp')"><h3>Регистрация</h3></button>
        </div>
        <!-- Tab content -->
        <form method="POST" action="">
          <div id="SignIn" class="tabcontent">
            <table>
              <tr>
                <td>Имя пользователя</td>
                <td><input type="text"
name="in_login" value="" class="field"></td>
              </tr>
              <tr>
                <td>Пароль</td>
                <td><input type="password"
name="in_password" value="" class="field"></td>
              </tr>
            </table>
            <p>
              <input type="submit" name="Submit" value="OK"
class="buttons">&nbsp;  
              <input type="reset" name="Reset"
value="Сброс" class="buttons">
            </p>
          </div>
          <div id="SignUp" class="tabcontent">
            <table>
              <tr>
                <td>Имя пользователя</td>
                <td><input type="text" name="login"
value="" class="field"></td>
              </tr>
              <tr>
                <td>E-mail</td>
                <td><input type="text" name="email"
value="" class="field"></td>
              </tr>
              <tr>
                <td>Пароль</td>
                <td><input type="password"
name="password" value="" class="field"></td>
              </tr>
              <tr>
                <td>Подтвердите пароль</td>
                <td><input type="password"
name="validate_password" value="" class="field"></td>
              </tr>
            </table>
          </div>
        </form>
      </div>
    </div>
  </div>
</div>

```

```

</table>
<p>


```

Код модального окна с вкладками (JavaScript)

```

<script>
function openEvent (evt, eventName) {
    var i, tabcontent, tablinks;
    tabcontent = document.getElementsByClassName("tabcontent");
    for (i = 0; i < tabcontent.length; i++) {
        tabcontent[i].style.display = "none";
    }
    tablinks = document.getElementsByClassName("tablinks");
    for (i = 0; i < tablinks.length; i++) {
        tablinks[i].className = tablinks[i].className.replace("
active", "");
    }
    document.getElementById(eventName).style.display = "block";
    evt.currentTarget.className += " active";
}
</script>

```

Код модального окна с вкладками (CSS)

```

.tab {
    overflow: hidden;
    background-color: #fff;
}
.tab button {
    background-color: inherit;
    float: left;
    border: none;
    outline: none;
    cursor: pointer;
    padding: 0px 90px;
    transition: 0.3s;
    font-family: Cuprum, Arial, Helvetica, sans-serif;
    background-color: #d7fafc;
    color: #00ced1;
}
.tab button:hover {
    background-color: #00ced1;
    color: #d7fafc;
}
.tab button.active {
    background-color: #00ced1;
    color: #d7fafc;
}
.tabcontent {

```

```

display: none;
padding: 6px 12px;
border-top: none;
}
.modal {
  position: fixed; /* фиксированное положение */
  top: 0;
  right: 0;
  bottom: 0;
  left: 0;
  background: rgba(0,0,0,0.5); /* цвет фона */
  z-index: 1050;
  opacity: 0; /* по умолчанию модальное окно прозрачно */
  -webkit-transition: opacity 200ms ease-in;
  -moz-transition: opacity 200ms ease-in;
  transition: opacity 200ms ease-in; /* анимация перехода */
  pointer-events: none; /* элемент невидим для событий мыши */
  margin: 0;
  padding: 0;
}
/* при отображении модального окна */
.modal:target {
  opacity: 1; /* делаем окно видимым */
  pointer-events: auto; /* элемент видим для событий мыши */
  overflow-y: auto; /* добавляем прокрутку по y, когда элемент не
помещается на страницу */
}
/* ширина модального окна и его отступы от экрана */
.modal-dialog {
  position: relative;
  width: auto;
  margin: 10px;
}
@media (min-width: 576px) {
  .modal-dialog {
    max-width: 500px;
    margin: 30px auto; /* для отображения модального окна по центру */
  }
}
/* свойства для блока, содержащего контент модального окна */
.modal-content {
  position: relative;
  display: -webkit-box;
  display: -webkit-flex;
  display: -ms-flexbox;
  display: flex;
  -webkit-box-orient: vertical;
  -webkit-box-direction: normal;
  -webkit-flex-direction: column;
  -ms-flex-direction: column;
  flex-direction: column;
  background-color: #fff;
  -webkit-background-clip: padding-box;
  background-clip: padding-box;
  border: 1px solid rgba(0,0,0,.2);
  border-radius: .3rem;
  outline: 0;
}
@media (min-width: 768px) {
  .modal-content {
    -webkit-box-shadow: 0 5px 15px rgba(0,0,0,.5);
    box-shadow: 0 5px 15px rgba(0,0,0,.5);
  }
}

```

```

}
/* свойства для заголовка модального окна */
.modal-header {
  display: -webkit-box;
  display: -webkit-flex;
  display: -ms-flexbox;
  display: flex;
  -webkit-box-align: center;
  -webkit-align-items: center;
  -ms-flex-align: center;
  align-items: center;
  -webkit-box-pack: justify;
  -webkit-justify-content: space-between;
  -ms-flex-pack: justify;
  justify-content: space-between;
  padding: 15px;
  border-bottom: 1px solid #eceeef;
}
.modal-title {
  margin-top: 0;
  margin-bottom: 0;
  line-height: 1.5;
  font-size: 1.25rem;
  font-weight: 500;
}
/* свойства для кнопки "Закрыть" */
.close {
  float: right;
  font-family: sans-serif;
  font-size: 24px;
  font-weight: 700;
  line-height: 1;
  color: #000;
  text-shadow: 0 1px 0 #fff;
  opacity: .5;
  text-decoration: none;
}
/* свойства для кнопки "Закрыть" при нахождении её в фокусе или наведении
*/
.close:focus, .close:hover {
  color: #000;
  text-decoration: none;
  cursor: pointer;
  opacity: .75;
}
/* свойства для блока, содержащего основное содержимое окна */
.modal-body {
  position: relative;
  -webkit-box-flex: 1;
  -webkit-flex: 1 1 auto;
  -ms-flex: 1 1 auto;
  flex: 1 1 auto;
  padding: 15px;
  overflow: auto;
}

```


Пример кода полей (HTML)

```
<input type=text name="doctype" value="" class="field">
<select name = 'fio' class = 'select'>
<textarea name="lettertext" cols = "40" rows="5" class="field"
style="width:100%"></textarea>
```

Код поля input (CSS)

```
.field {
    font-family: Cuprum, Arial, Helvetica, sans-serif;
    margin-left: 1rem;
    display: block;
    width: 150%;
    height: calc(1.25rem + 2px);
    padding: 0.375rem 0.75rem;
    font-family: inherit;
    font-size: 1rem;
    font-weight: 400;
    line-height: 1.5;
    color: #212529;
    background-color: #fff;
    background-clip: padding-box;
    border: 1px solid #00ced1;
    border-radius: 0.25rem;
    transition: border-color 0.15s ease-in-out, box-shadow 0.15s ease-
in-out;
}
```

Код поля select (CSS)

```
.select {
    font-family: Cuprum, Arial, Helvetica, sans-serif;
    margin-left: 1rem;
    display: block;
    width: 165%;
    height: 38px;
    padding: 0.375rem 0.75rem;
    font-family: inherit;
    font-size: 1rem;
    font-weight: 400;
    line-height: 1.5;
    color: #212529;
    background-color: #fff;
    background-clip: padding-box;
    border: 1px solid #00ced1;
    border-radius: 0.25rem;
    transition: border-color 0.15s ease-in-out, box-shadow 0.15s ease-
in-out;
}
```

Приложение 5

Пример кода кнопки (HTML)

```
<input class="buttons" type="submit" name="Submit" value="Далее">
```

Код кнопки (CSS)

```
.buttons {  
    background-color: #00ced1;  
    border: none;  
    color: white;  
    padding: 15px 32px;  
    text-align: center;  
    text-decoration: none;  
    display: inline-block;  
    font-size: 16px;  
    -webkit-transition-duration: 0.4s; /* Safari */  
    transition-duration: 0.4s;  
    font-family: Cuprum, Arial, Helvetica, sans-serif;  
}  
.buttons:hover {  
    background-color: #d7fafc;  
    color: #00ced1;  
}
```

Пример кода кнопки типа radio (HTML)

```
<input type="radio" name="on" value="8:30">
```

Код кнопки типа radio (CSS)

```
/* для элемента input с type="radio" */  
.custom-radio {  
    position: absolute;  
    z-index: -1;  
    opacity: 0;  
}  
/* для элемента label связанного с .custom-radio */  
.custom-radio+label {  
    display: inline-flex;  
    align-items: center;  
    user-select: none;  
}  
/* создание в label псевдоэлемента before со следующими стилями */  
.custom-radio+label::before {  
    content: '';  
    display: inline-block;  
    width: 1em;  
    height: 1em;  
    flex-shrink: 0;  
    flex-grow: 0;  
    border: 1px solid #9ae8ec;  
    border-radius: 50%;  
    margin-right: 0.5em;  
    background-repeat: no-repeat;  
    background-position: center center;  
    background-size: 50% 50%;  
}  
/* стили при наведении курсора на радио */  
.custom-radio:not(:disabled):not(:checked)+label:hover::before {
```

```

    border-color: #00ced1;
}
/* стили для активной радиокнопки (при нажатии на неё) */
.custom-radio:not(:disabled):active+label::before {
    background-color: #00ced1;
    border-color: #00ced1;
}
/* стили для радиокнопки, находящейся в фокусе */
.custom-radio:focus+label::before {
    box-shadow: 0 0 0 0.2rem rgba(0, 123, 255, 0.25);
}
/* стили для радиокнопки, находящейся в фокусе и не находящейся в
состоянии checked */
.custom-radio:focus:not(:checked)+label::before {
    border-color: #9ae8ec;
}
/* стили для радиокнопки, находящейся в состоянии checked */
.custom-radio:checked+label::before {
    border-color: #00ced1;
    background-color: #00ced1;
}
/* стили для радиокнопки, находящейся в состоянии disabled */
.custom-radio:disabled+label::before {
    background-color: #d7fafc;
}
}

```

Открытие сессии (PHP)

```
$start=session_start();
```

Сохранение данных в сессию (PHP)

```
$_SESSION['fio'] = $fio;
```

Получение данных из сессии (PHP)

```
$fio = $_SESSION['fio'];
```

Подключение к базе данных (PHP)

```
$link = mysqli_connect("localhost", "people", "12345", "medsys"); //
подключение к локальному серверу localhost, к базе данных medsys по имени
пользователя people и паролю 12345
$addition = mysqli_query($link, "SET NAMES 'cp1251';"); // установление
кодировки, в которой будут считываться данные, для корректного
отображения текста на русском языке
```

Примеры запросов к базе данных (PHP)

```
$que = mysqli_query($link, "select concat(users.surname, '
',users.firstname, ' ',users.lastname) as fio from users join doctors on
doctors.docname = users.username where doctors.doctype = '$doctype'");
```

```
$result = mysqli_query($link, "UPDATE medsys.users SET
firstname='$firstname', lastname='$lastname', surname='$surname',
email='$email', address='$address', phone='$phone',
birthday='$birthday' WHERE username='$username'");
```

```
$result = mysqli_query($link, "SELECT COUNT(*) AS count FROM
medsys.registration WHERE username = '$in_login' and password =
md5('$in_password')");
```

```
$adding = mysqli_query($link, "INSERT INTO medsys.registration
(username,password) VALUES ('$login',md5('$password'))");
```

Получение значений из assoc (PHP)

```
$object = mysqli_fetch_array($que); // преобразование assoc в array,
далее для получения значения по ключу пишется $object[ключ]
```

Вывод таблицы с результатами запроса (PHP, HTML)

```
// Выводим заголовок таблицы:
echo"<table border=\"1\" bordercolor=\"#00ced1\" width=\"50%\"
bgcolor=\"#d7fafc\">";
echo "<tr><td>Врач</td><td>Фамилия врача</td><td>Имя
врача</td><td>Отчество врача</td><td>Дата</td>";
echo "<td>Время</td>"; //td
// Выводим таблицу:
for ($c=0; $c<mysqli_num_rows($q); $c++)
{echo "<tr>";
$f = mysqli_fetch_array($q);
echo
"<td>$f[doctype]</td><td>$f[surname]</td><td>$f[firstname]</td><td>$f[las
tname]</td><td>$f[date]</td>";
echo "<td>$f[time]</td>";
echo "</tr>";
} //конец цикла вывода таблицы
echo "</table>";
```

Генерация письма (PHP)

```
$lettertext = $_POST["lettertext"]; // текст письма - из сессии
$addressee  = $_f_doc[email]; // кому отправляем - из результата запроса
$sub='письмо от пациента'; //тема письма
$email=$_f_user[email]; // от кого - из результата запроса
$send      = mail ($addressee,$sub,$lettertext,"Content-type:text/plain;
charset = utf-8\r\nFrom:$email");
```