**Министерство образования Иркутской области**

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

КР.09.02.07.22.201.11 ПЗ

ИНФОРМАЦИОННАЯ СИСТЕМА

«БОЛЬНИЦА. РЕГИСТРАТУРА»

|  |  |  |
| --- | --- | --- |
| Председатель ВЦК: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Кудрявцева) |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Кудрявцева) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (А.В. Макрицкий) |
|  |  |  |

Иркутск 2022

**Содержание**

[ВВЕДЕНИЕ 3](#_Toc122369592)

[1. Описание предметной области ИС 4](#_Toc122369593)

[2. Анализ инструментов 5](#_Toc122369594)

[3. Техническое задание 12](#_Toc122369595)

[4. Проектирование ИС 13](#_Toc122369596)

[4.1 Структурная схема ИС 13](#_Toc122369597)

[4.2 Функциональная схема ИС 15](#_Toc122369598)

[4.3 Проектирование базы данных 17](#_Toc122369599)

[4.4. Проектирование интерфейса 21](#_Toc122369600)

[5. Разработка ИС 23](#_Toc122369601)

[5.1. Разработка интерфейса ИС 23](#_Toc122369602)

[6. Технологическая документация ИС 31](#_Toc122369603)

[6.1 Руководство пользователя ИС 31](#_Toc122369604)

[ЗАКЛЮЧЕНИЕ 33](#_Toc122369605)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 34](#_Toc122369606)

[Приложение А – Техническое задание 35](#_Toc122369607)

[Приложение Б – Листинг roles.php 40](#_Toc122369608)

## **ВВЕДЕНИЕ**

Регистратура больницы является одним из важнейших структурных подразделений медицинского учреждения. В данный момент запись на прием к врачу является проблематичной из-за сложности работы с бумажными носителями, а также недостатка легких и понятных информационных средств регистрации. Компьютерные технологии автоматизируют информационные процессы лечебного учреждения, тем самым процесс регистрации становится более простым. В связи с этим, возникла необходимость в разработке такой информационной системы как программа для записи и хранения данных о пациенте, которая должна обеспечить поиск все о пациенте, какого врача он выбрал и др.

Цель курсовой работы – разработать информационную систему «Больница. Регистратура».

Задачи курсовой работы:

* Провести предпроектное исследование предметной области (выбрать тему, поставить цель и задачи, описать предметную область).
* Разработать техническое задание.
* Спроектировать программный продукт.
* Разработать и отладить программный продукт.
* Составить программную документацию в виде руководства пользователя.

Актуальность разрабатываемой программы заключается в том, что каждый желающий может прийти в больницу, подойти к электронной регистратуре и воспользоваться ей. Компьютеры помогают человеку в работе, развлечении, образовании, в научных исследованиях и др. Программа для записи данных о пациенте «Регистратура больницы» позволит исключить многолюдную очередь в регистратуру.

## **Описание предметной области ИС**

Предметной областью курсовой работы является регистратура в больнице. Регистратура - структурное подразделение медицинского учреждения, непосредственно осуществляющее медицинскую деятельность, по формам и технологиям здравоохранения.

Регистратура любой медицинской организации ведет учет и регистрацию населения, проживающего на участке, обсуживающимся данной поликлиникой. Оформляет первичный прием, заполняет больничные карточки, отслеживает ведение историй болезни и посещение больными узких специалистов, оформляет и выдает талоны на прием к узким специалистам. Формирует и отслеживает расписание работы узких специалистов. Выдает результаты анализов и больничные карточки посетителям поликлиники. Ведет архив больничных карточек.

В медицинских организациях работа с пациентами в регистратуре поликлиники ведется в ручном режиме, что приводит к потере времени и ненадлежащему контролю.

Выбранная тема исследования является актуальной в связи с внедрением автоматизированной системы регистрации пациентов, что значительно упростит и ускорит процесс обслуживания пациентов.

Информация, хранимая в базе данных информационной системы регистратуры: информация о врачах, сведения о приемах, информация о пациентах.

## **Анализ инструментов средств разработки**

Грамотно выбранные инструменты для создания программного продукта, заранее определяют его результат.

Проектировать структуру информационной системы удобно через MySQL Workbench и Draw.io, а дизайн – через сервис Figma. Для работников информационная система будет серверной. Для реализации клиентской части подойдут следующие инструменты: HTML5, CSS3 и JS, включая AJAX.

MySQL Workbench — это унифицированный визуальный инструмент для архитекторов баз данных, разработчиков и администраторов баз данных. MySQL Workbench предоставляет моделирование данных, разработку SQL и комплексные инструменты администрирования для настройки сервера, администрирования пользователей, резервного копирования и другое.

Draw.io — это бесплатное кроссплатформенное программное обеспечение для рисования графиков с открытым исходным кодом, разработанное на HTML5 и JavaScript. Его интерфейс можно использовать для создания таких диаграмм, как блок-схемы, каркасы, UML-диаграммы, организационные диаграммы и сетевые диаграммы.

Figma – графический редактор для создания прототипов сайтов и приложений. Над проектом одновременно могут работать несколько человек, так как можно выдать доступ на редактирование или комментирование любому. В Figma обычно создают прототипы сайтов и приложений, иллюстрации, векторную графику, рисуют элементы интерфейса. Ещё здесь создают макеты сайтов для тильды: есть возможность импортировать дизайн.

HTML – язык разметки гипертекста. Язык разметки дает браузеру необходимые инструкции о том, как отображать тексты и другие элементы страницы на мониторе. Язык HTML интерпретируется браузерами и отображается в виде документа, в удобной для человека форме.

CSS – каскадные таблицы стилей, которые используются для определения стилей (правил) оформления документов — включая дизайн, вёрстку и вариации макета для различных устройств и размеров экрана.

JavaScript – это мультипарадигменный язык программирования, который обычно применяется в качестве встраиваемого инструмента для программного доступа к различным объектам приложений. С точки зрения веб-разработки, без знаний этой технологии невозможно заниматься созданием современных интерактивных сайтов. Язык JS – это то, что «оживляет» разметку страниц (HTML) и пользовательский функционал (CMS) сайтов. С помощью этого языка реализуется возможность реакции страницы или отдельных ее элементов на действия посетителя. Сегодня JavaScript является базовым языком программирования для браузеров. Он полностью совместим с операционными системами Windows, Linux, Mac OS.

AJAX, или, более длинно, Asynchronous Javascript And Xml - технология для взаимодействия с сервером без перезагрузки страниц. За счет этого уменьшается время отклика и веб-приложение по интерактивности больше напоминает десктоп.

Информационная система будет содержать в себе информацию – её необходимо хранить, изменять, структурировать и использовать. Это реализуется благодаря базе данных. Были рассмотрены следующие варианты реализации СУБД: MySQL, MySQL Workbench, PostgreSQL.

MySQL — одна из наиболее используемых систем управления базами данных. MySQL управляет реляционными базами данных, то есть такими, в которых таблицы связаны между собой. MySQL работает по принципу клиент-сервер. Компьютер пользователя (клиент) отправляет запрос. Сервер баз данных его обрабатывает и предоставляет ответ. Именно поэтому часто можно услышать понятие MySQL-сервер. Это сервер, на котором хранится база данных. Система MySQL написана на языках программирования C и C++. Для работы MySQL используется язык структурированных запросов SQL.

[MySQL Workbench](https://www.mysql.com/products/workbench/) — универсальный инструмент для работы и управления базами данных. С его помощью можно создавать и редактировать таблицы и другие объекты, управлять доступом пользователей и полноценно администрировать БД. Workbench может пригодиться разработчикам, администраторам БД и архитекторам систем.

Несмотря на название, он может работать не только с MySQL, но и с другими СУБД: Oracle Database, Microsoft SQL Server, PostgreSQL и другими. Одна из главных особенностей Workbench — визуальное моделирование. Оно позволяет увидеть все таблицы и связи между ними в наглядном виде. Это намного упрощает проектирование базы данных.

PostgreSQL — это объектно-реляционная система управления базами данных (ORDBMS), наиболее развитая из открытых [СУБД](https://blog.skillfactory.ru/glossary/subd/) в мире. Имеет открытый исходный код и является альтернативой коммерческим базам данных. СУБД позволяет гибко управлять базами данных. С ее помощью можно создавать, модифицировать или удалять записи, отправлять транзакцию — набор из нескольких последовательных запросов на особом языке запросов SQL.

Для наглядности сравнения вариантов реализации базы данных была составлена таблица 1.

Таблица 1 – Сравнение средств реализации базы данных

|  |  |  |
| --- | --- | --- |
| Название СУБД | MySQL | PostgreSQL |
| Большое кол-во типов данных | + | + |
| Популярность | + | - |
| Отказоустойчивость | - | + |
| Не требует удаленного сервера | - | - |
| Простота использования | - | - |
| Портативность | - | - |

Таким образом, в качестве базы данных для будущего продукта была выбрана MySQL, так как она предоставляет весь необходимый функционал для разработки продукта, и при этом она проста в изучении и овладении.

PHP — это распространённый язык программирования общего назначения с открытым исходным кодом. PHP специально сконструирован для веб-разработок и его код может внедряться непосредственно в HTML.

Язык PHP обладает рядом неоспоримых преимуществ:

* Высокая скорость работы и, соответственно, общая производительность ресурсов.
* Бюджетность, экономичность. Найти специалиста не представляется проблемой, стоимость написания программ на php не высока.
* Простота освоения, простой синтаксис.
* Отличная совместимость и переносимость — php-коды работают одинаково хорошо с разными платформами.
* Набор текста кода и его редактирование можно осуществлять в любом текстовом или html-редакторе.

Таким образом, php будет более лучшим вариантом, ведь он более компактный и простой в освоении язык. Он вобрал все лучшие особенности таких популярных языков, как C, Java и Perl, а также его сообщество более активное.

Для разработки программного продукта рассмотрены следующие инструментальные средства разработки программных продуктов:

1. PHPStorm.

2. Netbeans.

3. Atom.

PhpStorm — коммерческая кросс-платформенная [интегрированная среда разработки](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) обеспечивает первоклассную поддержку PHP 5.3–8.1, мгновенно предотвращает ошибки, предоставляет точное автодополнение и безопасные рефакторинги, а также возможность редактирования кода на HTML, CSS и JavaScript.

IDE обеспечивает умное автодополнение, подсветку синтаксиса, расширенные настройки форматирования, проверку ошибок на лету, сворачивание фрагментов кода, поддержку языковых комбинаций и много других возможностей для продуктивной разработки. Автоматические рефакторинги позаботятся о порядке в вашем коде и помогут легко и безопасно выполнять глобальные настройки проекта.

Сотни встроенных инспекций проверяют код прямо в режиме редактирования, а также позволяют проанализировать весь проект на наличие ошибок и проблем в структуре кода. Для большинства инспекций доступны быстрые исправления, которые позволяют откорректировать код мгновенно. Нажмите Alt+Enter, чтобы просмотреть список возможных исправлений для любой инспекции. PhpStorm помогает быстро перемещаться по коду и экономить время при работе над большими проектами. В один клик можно перейти к определению любого метода, функции или переменной, или найти их использования в проекте.

NetBeans IDE — свободная интегрированная среда разработки приложений (IDE) на языках программирования Java, Python, PHP, JavaScript, C, C++, Ада и ряда других. Проект NetBeans IDE поддерживается и спонсируется компанией Oracle, однако разработка NetBeans ведётся независимым сообществом разработчиков-энтузиастов (NetBeans Community) и компанией NetBeans Org. Последние версии NetBeans IDE поддерживают рефакторинг, профилирование, выделение синтаксических конструкций цветом, автодополнение набираемых конструкций на лету и множество предопределённых шаблонов кода. Для разработки программ в среде NetBeans и для успешной инсталляции и работы самой среды NetBeans должен быть предварительно установлен Sun JDK или J2EE SDK подходящей версии. Среда разработки NetBeans по-умолчанию поддерживала разработку для платформ J2SE и J2EE. Начиная с версии 6.0 NetBeans поддерживает разработку для мобильных платформ J2ME, C++ (только g++) и PHP без установки дополнительных компонентов.

Atom (в прошлом Atomicity) — бесплатный текстовый редактор с открытым исходным кодом для Linux, macOS, Windows с поддержкой плагинов, написанных на JavaScript, и встраиваемых под управлением Git. Большинство плагинов имеют статус свободного программного обеспечения, разрабатываются и поддерживаются сообществом.

Atom основан на Electron (ранее известный как Atom Shell) — фреймворке кроссплатформенной разработки с использованием Chromium и io.js. Редактор написан на CoffeeScript и LESS. Версия 1.0 была выпущена 25 июня 2015 г.

Сравнение IDE для разработки программного продукта наглядно представлено в таблице 2.

Таблица 2 – Сравнение IDE для разработки программного продукта

|  |  |  |  |
| --- | --- | --- | --- |
| Название IDE | PHPStorm | NetBeans | Atom |
| Распространяется бесплатно | + - | + | + |
| Автоматическое сохранение | + | - | + |
| Подсказки по коду | + | + | + |
| Интеграция с системой контроля версия (GIT) | + | + | - |
| Возможность расширения функционала библиотеками | + | + | + |
| Заточен под PHP-разработку | + | + | + |
| Поддержка CSS/HTML/JS | + | + | + |
| Комфортное использование на слабых ПК | + | - | + |

Таким образом, после рассмотрения вариантов средств разработок, было принято решение использовать PHPStorm. PHPStorm намного функциональнее, чем приведённые выше Atom и NetBeans. В PHPStorm имеется автоматическое сохранение, которое не позволит случайно потерять все наработки, а также он более удобен в работе.

Обоснование выбора

Для создания программного продукта было решено использовать

средства:

1) Для создания структурных схем, контекстной и диаграмм декомпозиции использовались CASE-средства – Draw.io.

2) Для наглядного составления структуры базы данных использовался инструмент для визуального проектирования баз данных, их редактирования и полного администрирования – MySQL Workbench.

3) Для разработки дизайна web-приложения использовался онлайн-сервис для разработки дизайна и прототипа сайта – Figma.

4) На этапе разработки программного продукта использовались инструменты и среды для разработки: PHP, Laravel, HTML5, CSS3, JavaScript и AJAX, а также редактор кода PHPStorm.

Основные преимущества веб-технологий:

1. Распределенность (пользователь может работать с системой из любого  
   места, связанного с WEB-сервером по сети, находясь в любой точке земного шара);
2. Переносимость (Web-клиенты (браузеры) существуют для любых  
   платформ, от настольных компьютеров до сотовых телефонов. Web-сервера используются для большинства платформ, а Web-приложения обычно пишутся на переносимых языках);  
   привычность интерфейса (почти каждый пользователь компьютера хотя бы раз запускал браузер и работал в нем);  
   простота установки и обслуживания (новую версию web-приложения не надо устанавливать на все компьютеры – достаточно установить на сервер).
3. Простота интерфейса - Пользователи не любят гигантских окошек с  
   сотнями полей ввода, а программисты и разработчики интерфейсов  
   почему-то любят. Web как раз не поощряет сложный интерфейс,  
   скорее поощряет простой.
4. Простота программирования интерфейса - Создавать HTML из  
   шаблонов куда проще и удобнее, чем создавать графические  
   приложения визуальными редакторами интерфейсов.
5. Простота программирования в целом - 3х-уровневая архитектура:  
   база данных – логика приложения (сервер) – логика представления  
   (клиент, браузер)

## **Техническое задание**

Техническое задание (далее ТЗ) — это документ, в котором фиксируются требования к проекту. Условно ТЗ можно назвать любое поручение исполнителю, главное, чтобы в нем были ясно прописаны характеристики итогового продукта.

ТЗ представлено в Приложении А.

## **Проектирование ИС**

## **4.1 Структурная схема ИС**

Проектирование информационной системы началось с построения диаграммы вариантов использования. На рисунке 1 представлена диаграмма прецедентов Uses CASE. Она содержит 2 актёра, которые могут выполнять суммарно 6 функций.

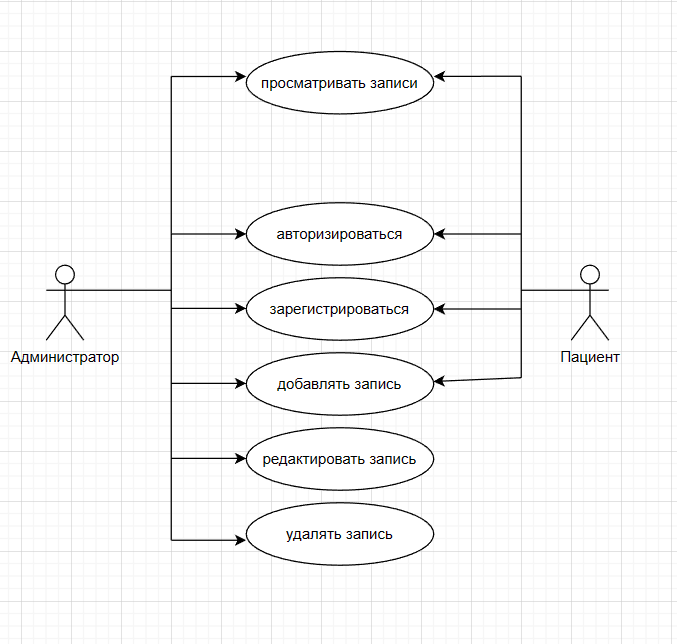


Рисунок 1 - Диаграмма прецедентов

На рисунке 2 представлена диаграмма деятельности. Она содержит 3 роли. Процесс начинается с пациента, который добавляет запись. Далее регистратура проверяет правильность записи, если запись возможна, то данные уходят к врачу, если нет, то пациент заново записывается.

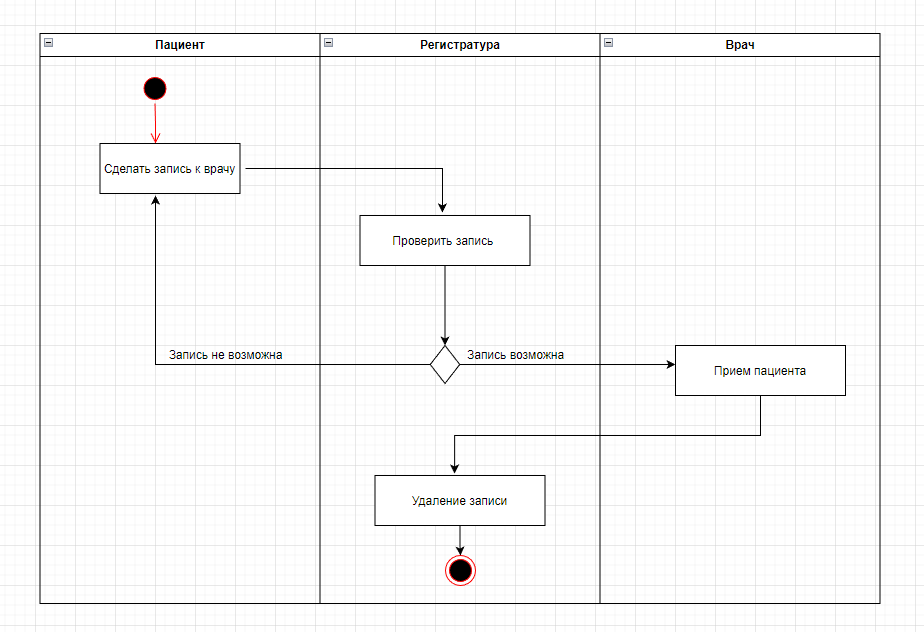


Рисунок 2 – Диаграмма деятельности

На рисунке 3 представлена диаграмма компонентов. В общей сложности 7 элементов.

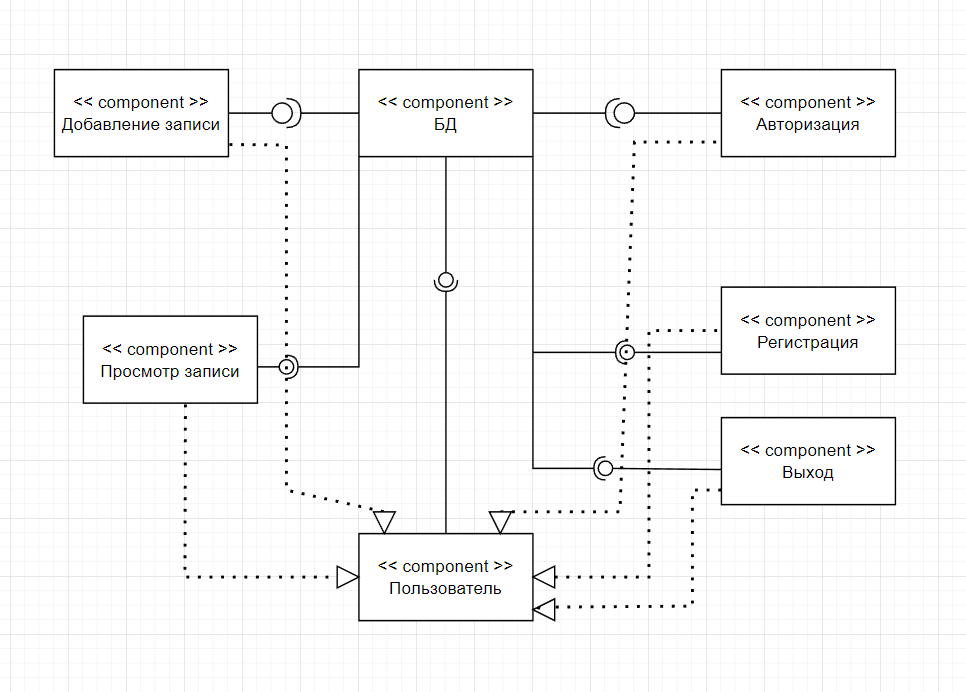


Рисунок 3 – Диаграмма компонентов

На рисунке 4 представлена диаграмма развёртывания. Она показывает, что, чтобы пользоваться программным продуктом, необходим web-сервер, на котором размещаются ИС и БД.

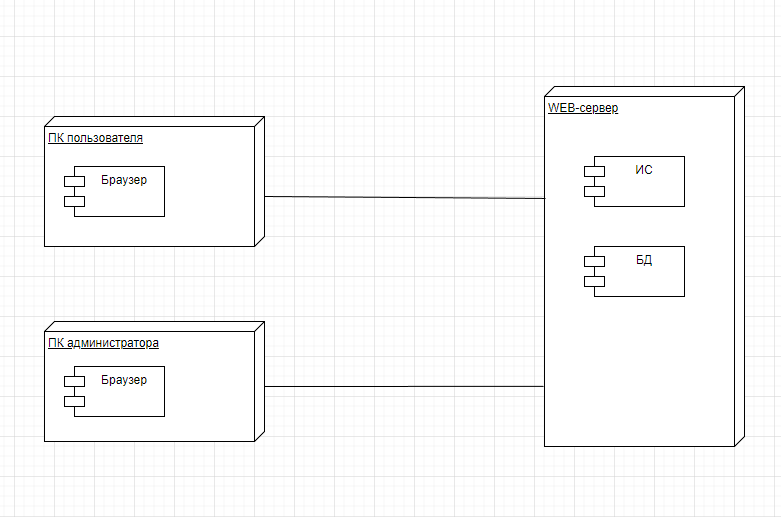


Рисунок 4 – Диаграмма развёртывания

## **4.2 Функциональная схема ИС**

На рисунке 5 представлена контекстная диаграмма, отображающая деятельность ИС.

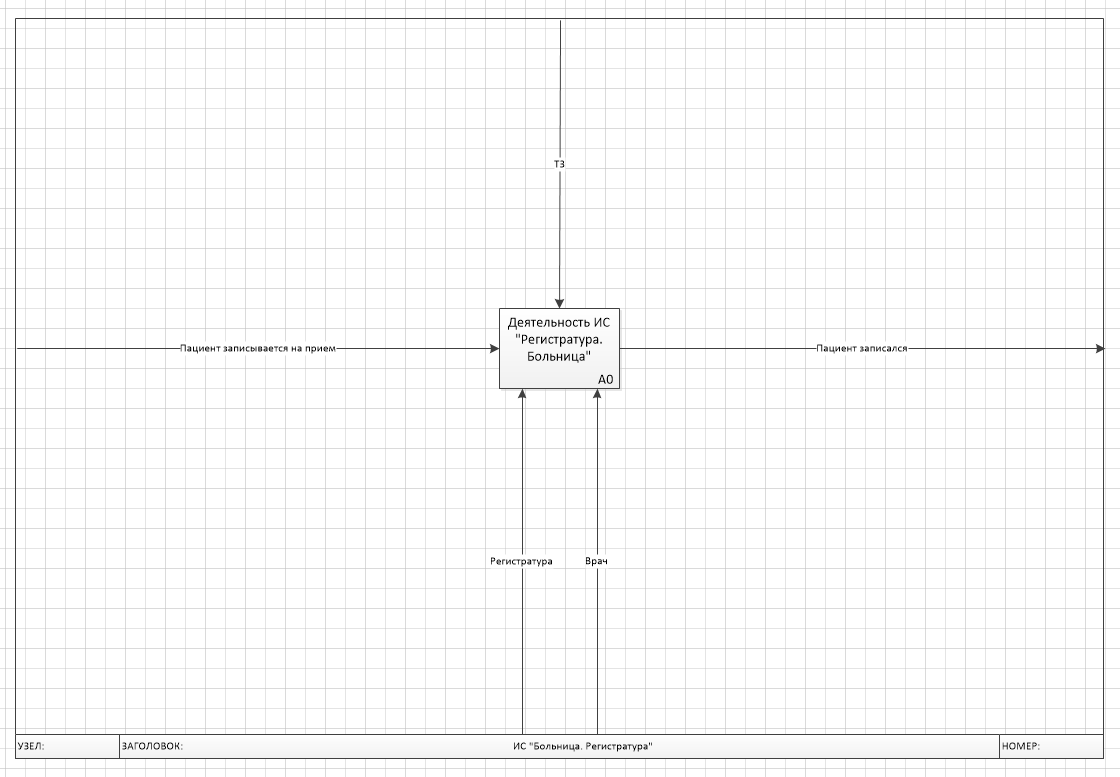


Рисунок 5 – Контекстная диаграмма IDEF0

На рисунке 6 представлена диаграмма декомпозиций (А1), отображающая деятельность ИС более подробно предыдущей.

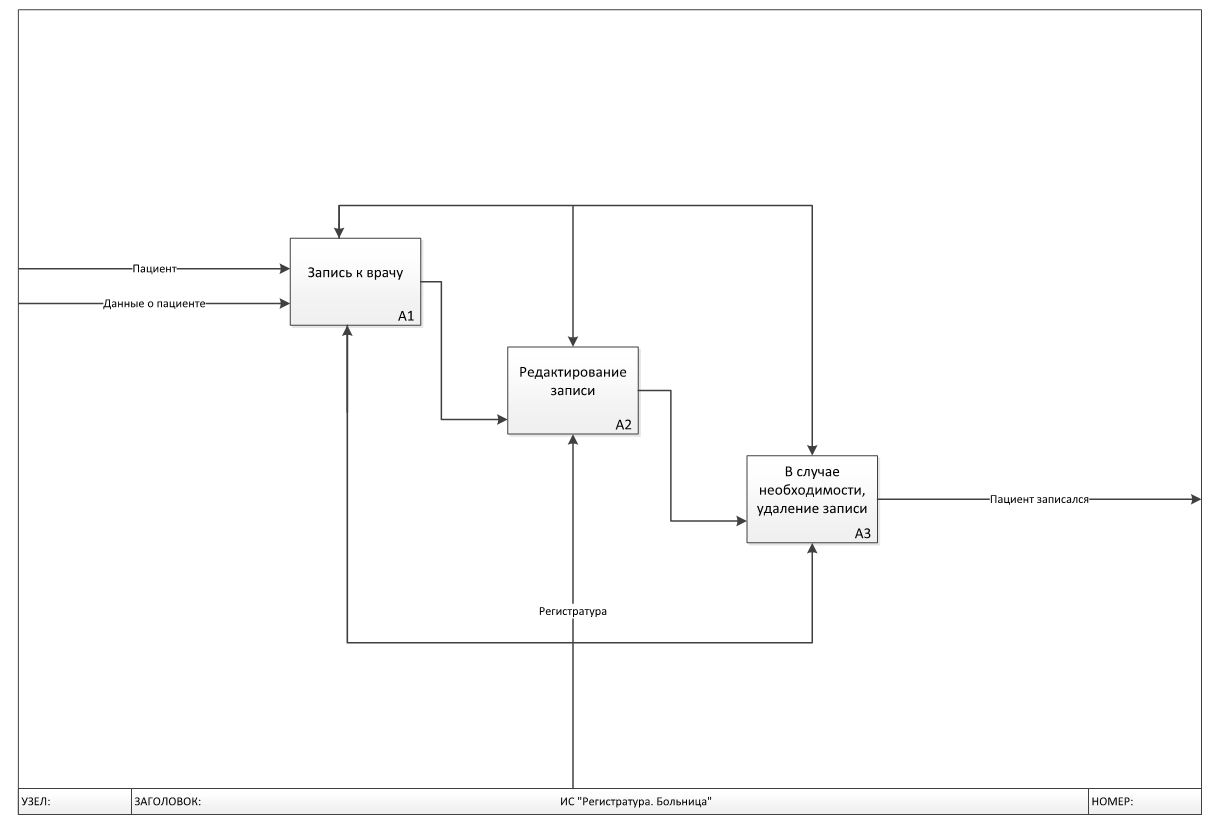
****

Рисунок 6 – Диаграмма декомпозиций А1

На рисунке 7 представлена диаграмма классов. Она содержит классы, среди них база данных, html-страница, информационная сеть «Регистратура. Больница», пользователь, врач, администратор, услуги.

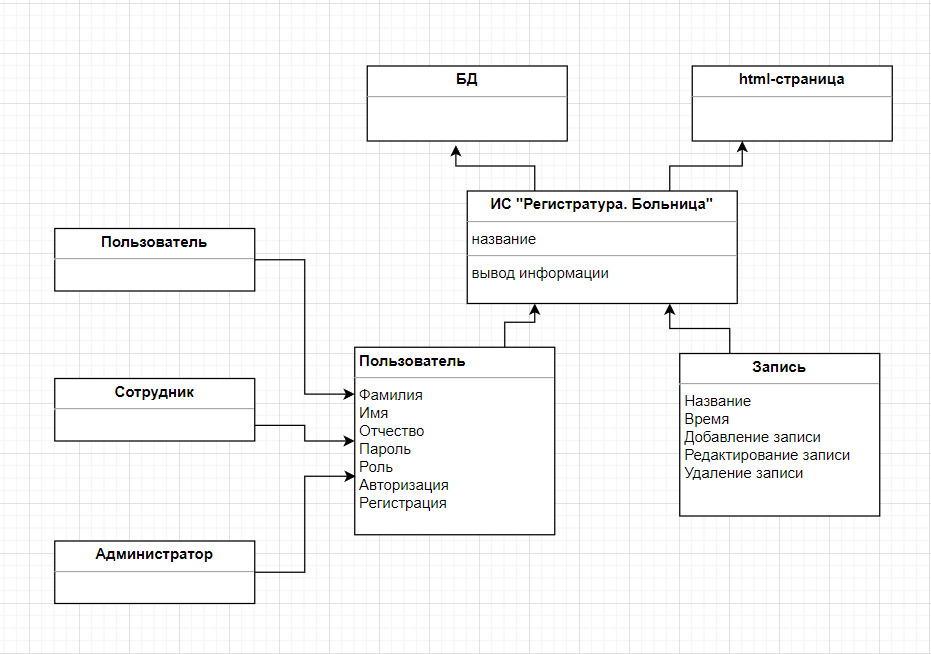


Рисунок 7 – Диаграмма классов

На рисунке 8 представлена диаграмма потоков данных. В центре всего пользователь, который может регистрироваться, просматривать услуги, добавлять, редактировать и удалять заказы. Для соответствующих действий в ИС предусмотрены соответствующие БД.

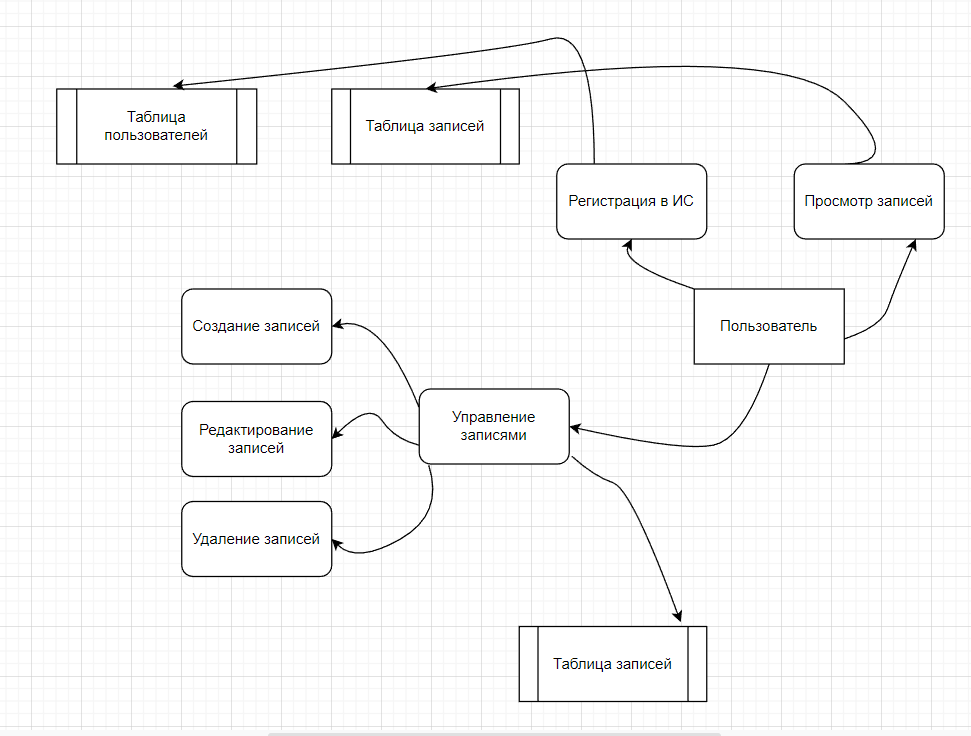


Рисунок 8 – Диаграмма потоков данных

## **Проектирование базы данных**

Проектирование базы данных начинается с концептуального проектирование базы данных.

Концептуальное проектирование – построение семантической модели предметной области, то есть информационной модели наиболее высокого уровня абстракции. Такая модель создаётся без ориентации на какую-либо конкретную СУБД и модель данных.

На рисунке 9 представлена инфологическая модель базы данных.

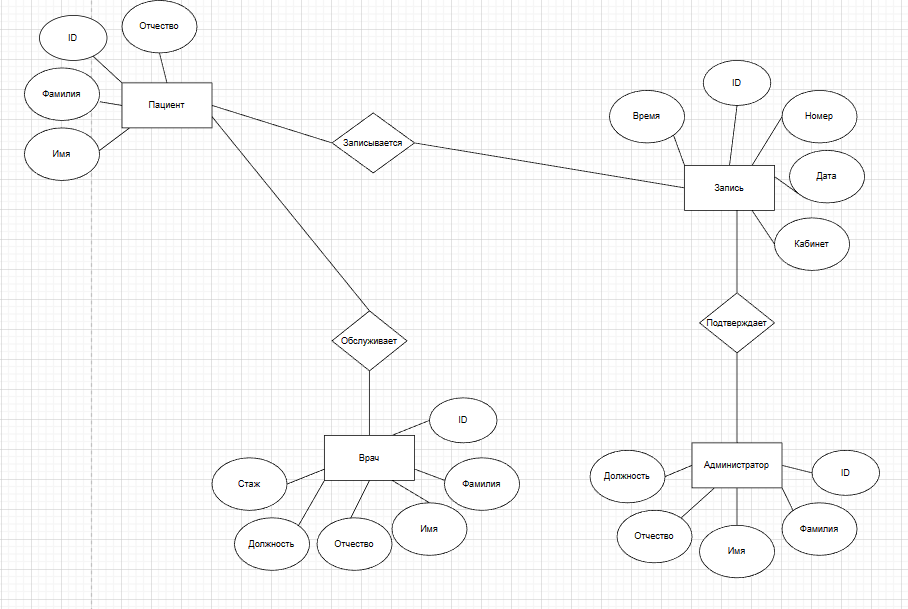
****

Рисунок 9 – Инфологическая модель базы данных

На инфологической модели базы данных выше схематично отображены сущности системы, их атрибуты и связи между ними. Так, в прямоугольник отображены сущности, такие как: Пациент, Администратор, Врач,Запись. В овалах отображены атрибуты сущностей, например: ФИО, ID и другие. Ромбами изображены связи между сущностями.

Далее происходит преобразование концептуальной модели в логическую модель, по формальным правилам. На даталогической модели базы данных (рисунок 10) отображены сущности приложения, а также первичные и внешние ключи, связывающие сущности между собой.

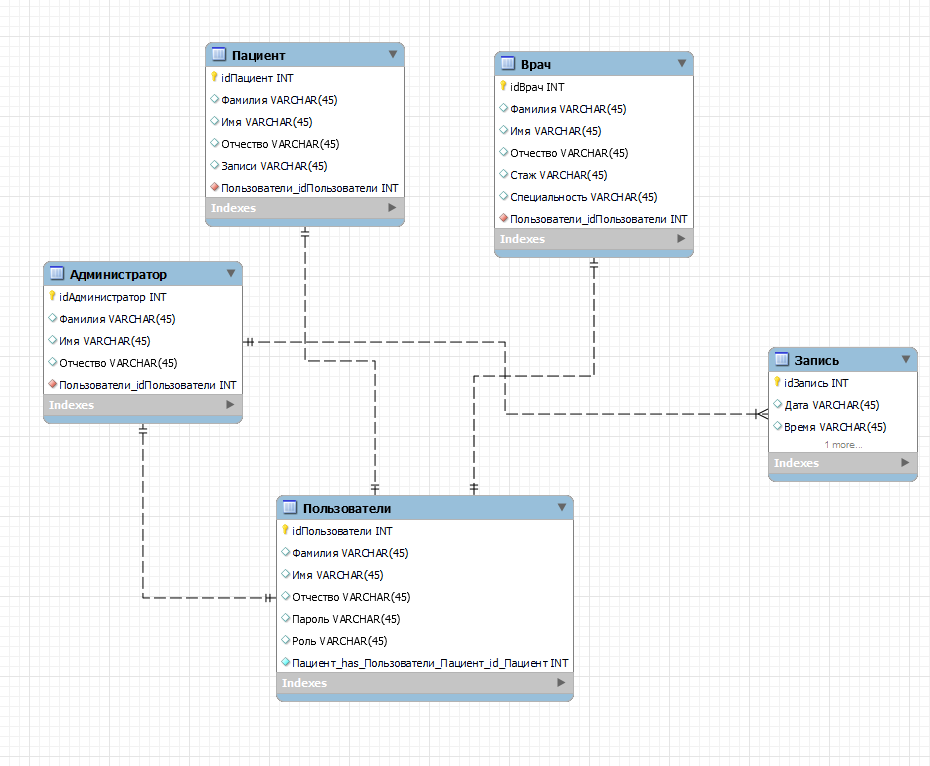


Рисунок 10 – Даталогическая модель базы данных

На рисунке 11 представлена ER-модель базы данных. Она содержит 6 таблиц для полного функционирования и качества сортировки информации.

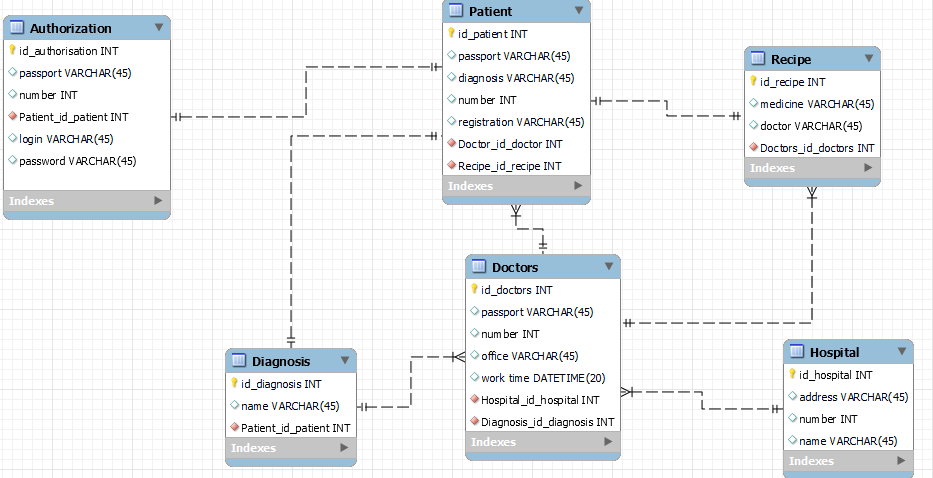
****

Рисунок 11 - ER-модель базы данных

Далее представлен перечень таблиц 3-7.

Таблица 3 – Таблица Services

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| id\_services | INT | Id услуги |
| Deadlines | DATE | Крайний срок |
| Service | VARCHAR(100) | Обслуживание |
| Staff\_id | INT | Id персонала |

Таблица 4 – Таблица Patient

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| id\_patient | INT | id пациента |
| Last\_name | VARCHAR(45) | Фамилия |
| Name | VARCHAR(45) | Имя |
| Father\_name | VARCHAR(45) | Отчество |
| Id\_orders | INT | Id записи |

Таблица 5 – Таблица Orders

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| Id\_orders | INT | Id записи |
| start | date | Время |
| name | VARCHAR(100) | Название |

Таблица 6 – Таблица Staff

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| Id\_staff | INT | Id персонала |
| Last\_name | VARCHAR(45) | Фамилия |
| Name | VARCHAR(45) | Имя |
| Father\_name | VARCHAR(45) | Отчество |

Таблица 7 – Таблица Users

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| Id\_users | INT | Id пользователя |
| Last\_name | VARCHAR(45) | Фамилия |
| Name | VARCHAR(45) | Имя |
| Password | VARCHAR(45) | Отчество |
| Role | INT | Роль |

## **4.4. Проектирование интерфейса**

Для разработки пользовательского интерфейса был выбран инструмент Figma — онлайн-сервис для разработки интерфейсов и прототипирования с возможностью организации совместной работы в режиме реального времени.

На рисунке 12 показан вход в личный кабинет.

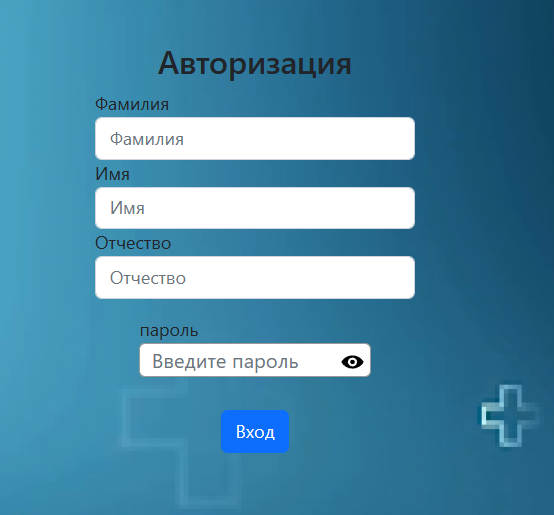
****

Рисунок 12 – Вход в личный кабинет

На рисунке 13 изображен раздел «Услуги»

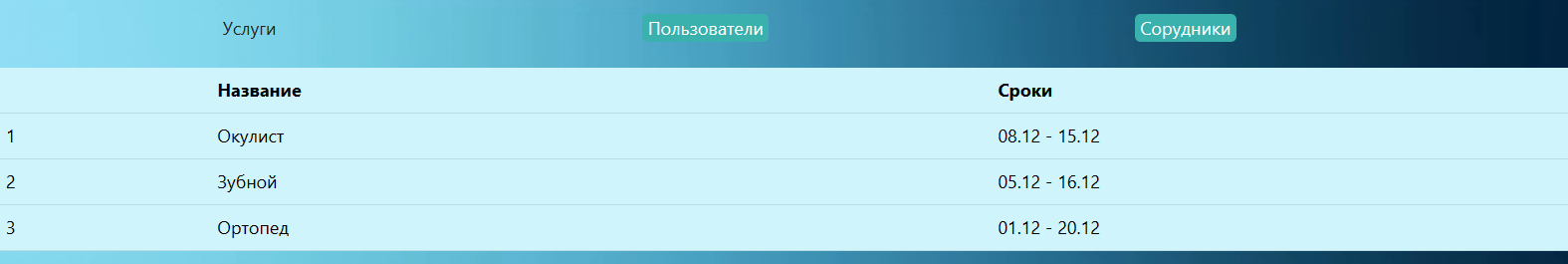
****

Рисунок 13 – Раздел «Врачи»

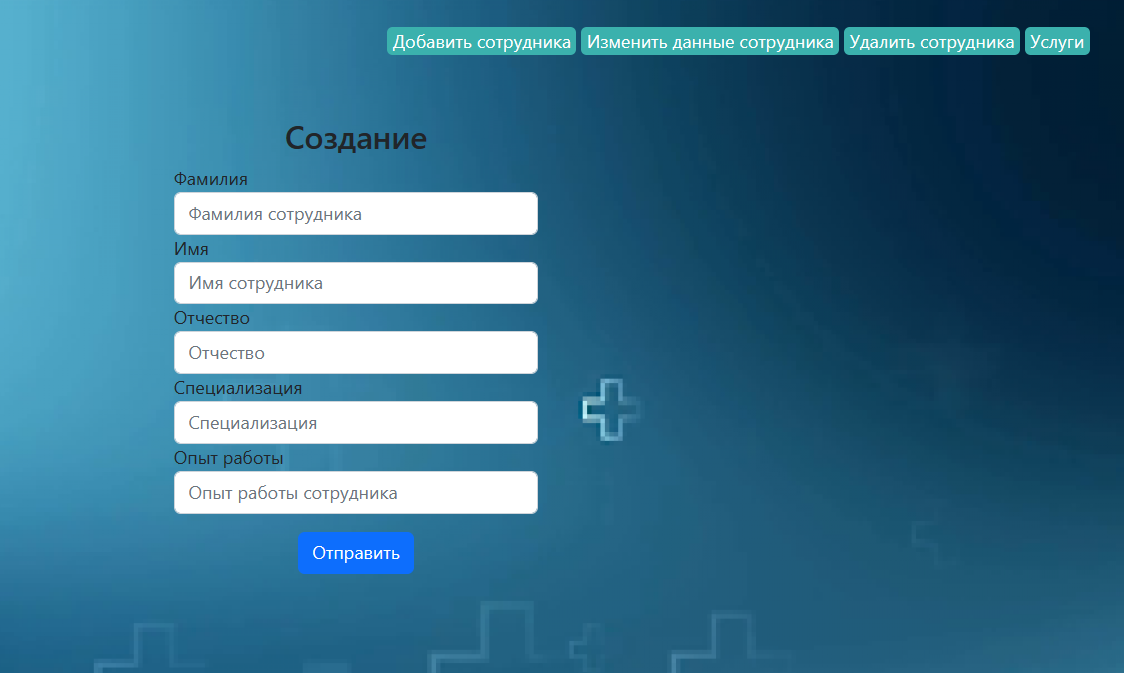
****

Рисунок 14 – Раздел «Добавление сотрудников»

## **Разработка ИС**

## **5.1. Разработка интерфейса ИС**

В информационной системе «Больница. Регистратура» были поставлены и выполнены следующие задачи:

− авторизация;

− регистрация;

− создание записей;

− просмотр текущих клиентов;

− удаление записей;

− редактирование записей;

− создание сотрудников;

− удаление сотрудников;

− редактирование сотрудников.

Программный продукт прост в освоении, т.к. имеет интуитивный интерфейс, в котором разобраться не составит труда даже начинающему пользователю ИС.

Для серверной операционной системы, на которой будет располагаться программный код ИС, выдвигаются следующие минимальные требования:

- процессор: Intel Pentium 4 или более поздней версии;

- ОЗУ 512 Мб DDR3;

- 15 ГБ на HDD;

- поддержка БД MySQL;

- операционная система Windows;

- доступ к сети Интернет.

Требования к клиентской части ИС, необходимые для пользования информационной системой:

- процессор Intel Pentium 4 или более поздней версии;

- ОЗУ 2 ГБ.

- доступ к сети Интернет;

- HDD 1ГБ;

- Браузер;

- встроенный видеоадаптер;

- клавиатура, мышь;

- операционная система: Windows.

Общий размер дисковой памяти, занимаемой информационной системой, составляет 3,5 МБ (рисунок 15).



Рисунок 15 – Объём дисковой памяти ИС

На рисунках 16 - 17 показан html код главной страницы. На ней находится лого, которое играет роль кнопки «Главная страница», кнопка «Авторизация» форма подключена к файлу header.php (в нем подключены стили css) и Services.php(в нем связь с таблицей services) к которому подключена форма и из которого осуществляется вывод данных из бд.



Рисунок 16 – Первая часть кода

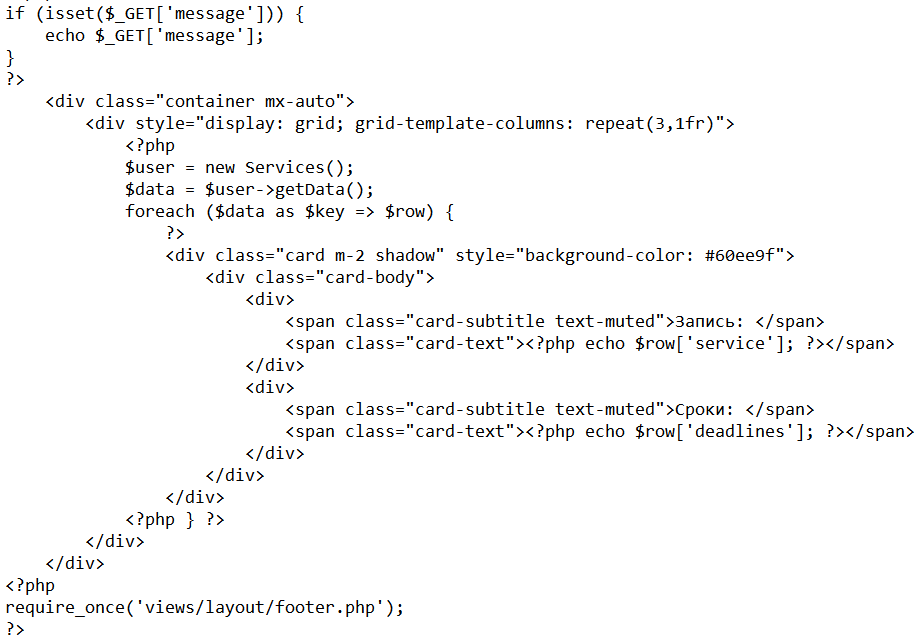


Рисунок 17 – Вторая часть кода

На рисунке 18 изображён результат создания главной страницы.

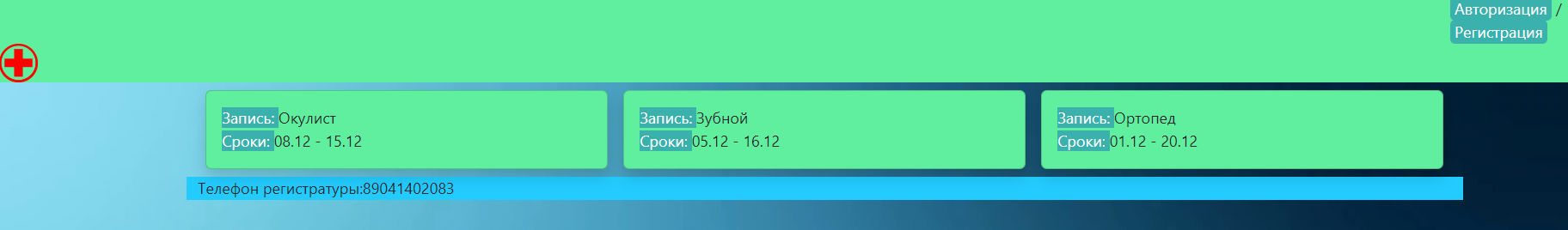


Рисунок 18 – Главная страница

На рисунках 19 и 20 изображен html код страницы авторизации.



Рисунок 19 – Первая часть кода страницы авторизации

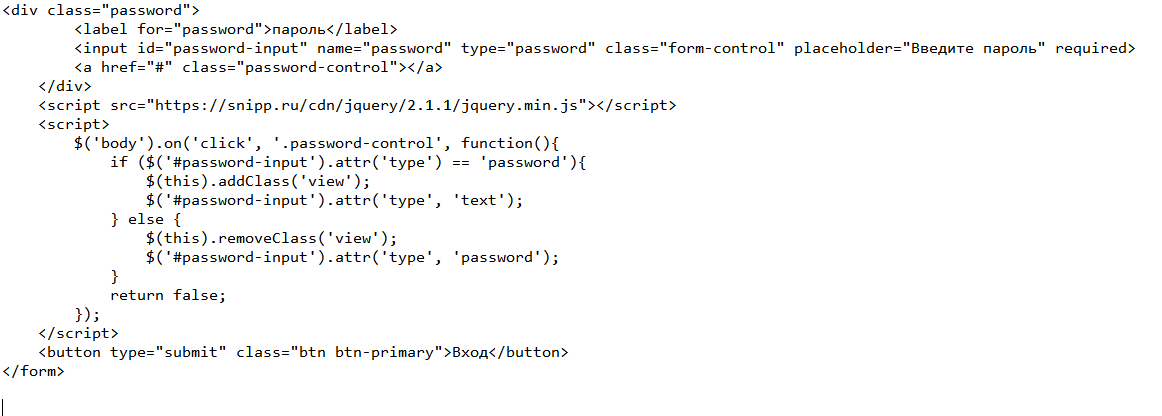


Рисунок 20 – Вторая часть кода страницы авторизации

На рисунке 21 изображен результат создания страницы авторизации.

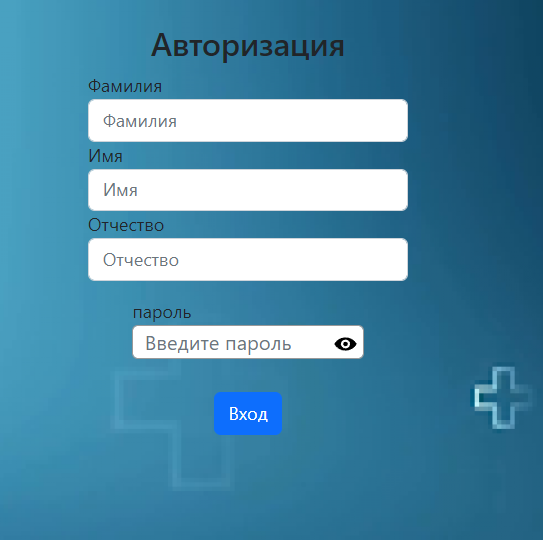


Рисунок 21 – Результат кода страницы авторизация

На рисунках 22 и 23 изображен html код страницы регистрации.



Рисунок 22 – Первая часть кода страницы регистрация

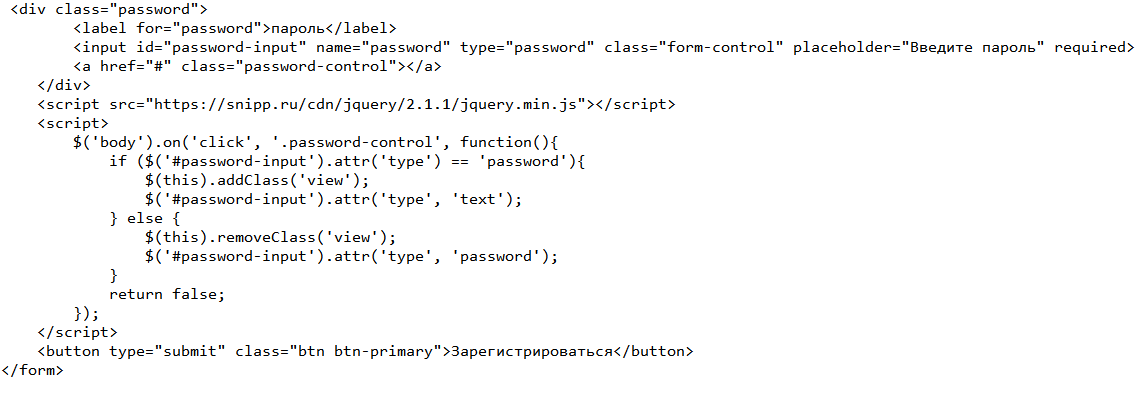


Рисунок 23 – Вторая часть кода страницы регистрация

На рисунке 24 изображен результат создания страницы регистрация

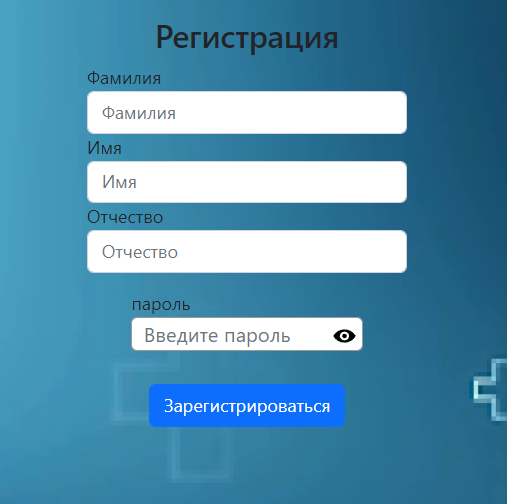


Рисунок 24 – Результат создания страницы регистрация

Так же было создано 2 личных кабинета (далее – ЛК) общего вида: для администратора для пользователя.

На рисунках 25 и 26 изображён html ЛК администратора.



Рисунок 25 – Первая часть кода ЛК администратора



Рисунок 26 – Вторая часть кода ЛК администратора



Рисунок 27 – Результат кода для ЛК администратора

На рисунках 28 и 29 изображён html ЛК пользователя.



Рисунок 28 – Первая часть кода ЛК пользователя



Рисунок 29 – Вторая часть кода ЛК пользователя

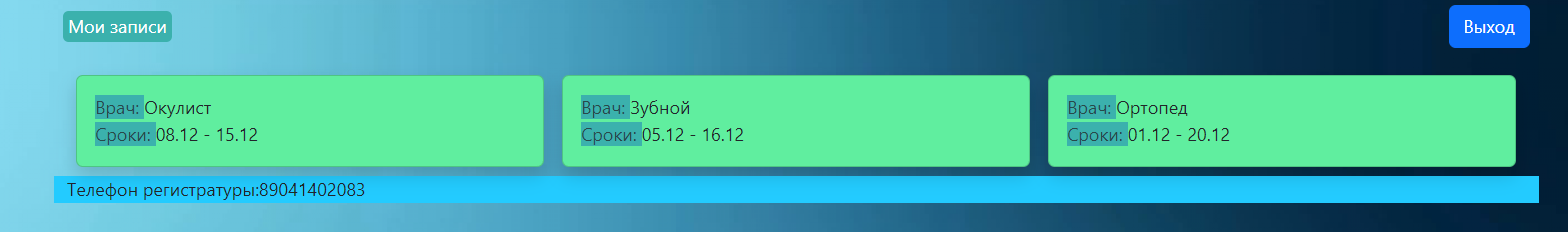
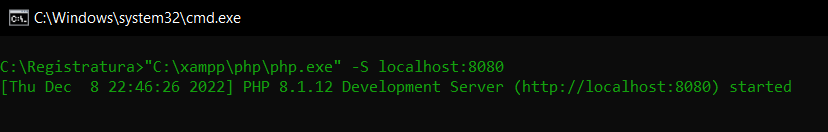


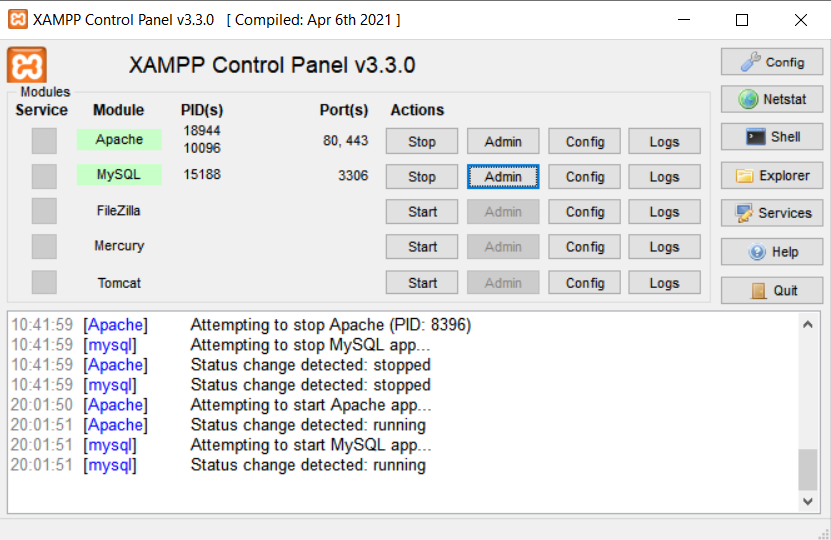
Рисунок 30 – Результат кода ЛК пользователя

## **Технологическая документация ИС**

## **6.1 Руководство пользователя ИС**

Для запуска данной информационной системы необходимо открыть файл start.bat и скопировать адрес сайта, далее вставить в адресную строку браузера. Так же нужно запустить XAMPP и запустить Apache и MySQL.





В данном руководстве пользователя рассмотрен пример Личный кабинет (Далее – ЛК) пользователя. На рисунке 31 можно видеть список врачей, а так же 2 кнопки: «Выход» и «Мои записи».

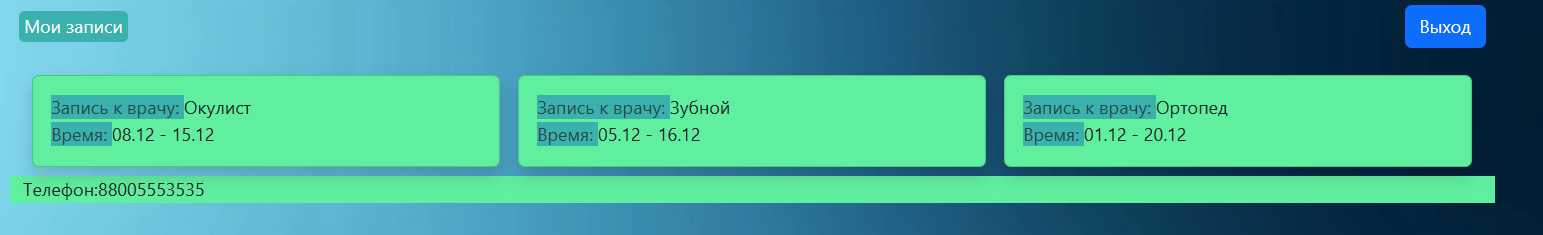


Рисунок 31 – ЛК пользователя

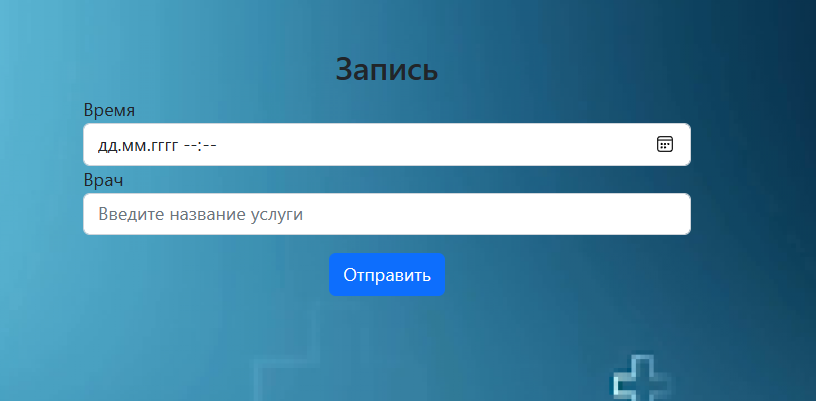


Рисунок 32 – Запись к врачу

## **ЗАКЛЮЧЕНИЕ**

В результате выполнения данной курсовой работы была разработана информационная система «Больница. Регистратура», которая позволяет упорядочить посещение врача. Был определен и реализован следующий функционал информационной системы:

− авторизация;

− регистрация;

− создание записей;

− просмотр текущих клиентов;

− удаление записей.

− редактирование записей.

− создание сотрудников;

− удаление сотрудников.

− редактирование сотрудников.

Были рассмотрены возможные реализации ИС с использованием разных технологий и языков программирования, но в результате анализа инструментальных средств разработки выбор остановился на языке программирования PHP.

Был разработан браузерный программный продукт, с интуитивным и современным интерфейсом и широким функционалом.

В дальнейшем ИС может развиваться путём расширения функционала и совершенствования интерфейса, тем самым набирая большее сообщество пользователей, а также и актуальность программного продукта.

Все поставленные в данном курсовом проекте цели и задачи были успешно выполнены.

## **СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1 Dmosk – Примеры SQL-запросов в MariaDB (MySQL) – URL: https://www.dmosk.ru/miniinstruktions.php?mini=sql-mysql (дата обращения: 11.10.2022). – Текст: электронный.

2 FrontEnd Resource – 107 Beautiful CSS Cards examples to improve your UI – URL: https://frontendresource.com/css-cards/(дата обращения: 21.11.2022). – Текст: электронный.

3 HTML5 BOOK – Основы CSS – URL: https://html5book.ru/osnovy-css/ (дата обращения:03.10.2022). – Текст: электронный. 4. Htmlbook – Самоучитель HTML

4 – URL: http://htmlbook.ru/samhtml (дата обращения: 01.10.2022). – Текст: электронный.

5 PHP – Объекты данных PHP – URL: https://www.php.net/manual/ru/book.pdo.php (дата обращения: 29.09.2022). – Текст: электронный.

6 sql academy – Интерактивный учебник по SQL – URL: https://sql-academy.org/ru/guide(дата обращения: 10.11.2022). – Текст: электронный.

7 old.code – Работа с сессиями PHP – URL: http://old.code.mu/books/php/auth/rabota-s-sessiyami-php.html(дата обращения: 10.11.2022). – Текст: электронный.

8 Stackoverflow – Реализовываем роли доступа на PHP сайте – URL: https://ru.stackoverflow.com/questions/231834/Реализовываем-роли-доступа-на-php-сайте(дата обращения: 14.11.2022). – Текст: электронный.

## 

## **Приложение А – Техническое задание**

**Министерство образования Иркутской области**

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

ИНФОРМАЦИОННАЯ СИСТЕМА

«БОЛЬНИЦА. РЕГИСТРАТУРА»

|  |  |  |
| --- | --- | --- |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Кудрявцева) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (А.В. Макрицкий) |
|  |  |  |

Иркутск 2022

**1. Общие сведения**

Наименование работы: информационная система «Регистратура. Больница».

Исполнитель: студент Иркутского Авиационного техникума, группы ИС-20-1, Макрицкий Андрей Владимирович

Разработка информационной системы проходит в рамках курсовой работы.

Сроки разработки информационной системы с 15.09.2022 по 06.12.2022 года.

**2. Цели и назначение создания автоматизированной системы**

Целью курсовой работы является создание информационной системы «Регистратура. Больница», которое включает в себя возможность записи на прием.

В данной информационной реализуется следующий функционал:

– ­авторизация;

– регистрация;

– запись на прием;

– просмотр записей;

– удаление записей.

**3. Характеристика объекта автоматизации**

Данная информационная система разрабатывается для облегчения записи в медицинское учреждение. При помощи сайта будет проще записаться к врачу.

**4. Требования к системе в целом**

**4.1 Требования к структуре и функционированию сайта**

Функции информационная система:

1. Окно «Авторизации»:
   1. авторизация пользователей.
2. Окно «Регистрация»:
   1. регистрация пользователей.
3. Окно «Главная страница»:
   1. Информация о записях;
4. Окно «Услуги»:
   1. Таблица «Услуги»:
      1. добавление записи;
      2. удаление записи;
      3. редактирование данных.
   2. Окно «Пациенты»:
      1. добавление пациентов;
      2. удаление пациентов;
      3. редактирование данных пациентов.
   3. Окно «Врачи»:
      1. добавление сотрудников;
      2. удаление сотрудников;
      3. редактирование данных.

**4.2 Требования к надежности**

Для обеспечения надежности необходимо проверять корректность получаемых данных. Входные данные поступают в виде значений c клавиатуры. Эти значения отображаются в отдельных полях таблицы.

**4.3 Требования к безопасности**

Для обеспечения безопасности в информационной системе, необходимо реализовать разграничение прав доступа.

**4.4 Требования к эксплуатации, техническому обслуживанию, ремонту и** **хранению компонентов системы**

Минимальные системные требования для сервера:

1. Процессор: Intel Pentium 4 2.0Ghz / AMD XP 2200+;
2. Оперативная память: 512 Мб;
3. Жёсткий диск: 150мб;
4. Операционная система: Windows 7/8/10.
5. Версия MySQL 5.0 и выше;

Минимальные системные требования для рабочей станции:

1. Процессор: Intel Pentium 4 2.0Ghz / AMD XP 2200+;
2. Оперативная память: 512 Мб;
3. Жёсткий диск: 150мб;
4. Операционная система: Windows 7/8/10.

**5. Состав и содержание работ по созданию системы**

В таблице 8 представлены плановые сроки начала и окончания работы по созданию веб-приложения.

Таблица 8 – Этапы выполнения разработки.

|  |  |  |
| --- | --- | --- |
| Наименование этапов разработки | Срок | |
| Начало | Окончание |
| Предпроектное исследование предметной области (выбор темы, постановка цели, задач, описание области применения, исследование предметной области) | 15.09.22 | 21.09.22 |
| Разработка технического задания (выбор архитектуры программного обеспечения, выбор типа пользовательского интерфейса, выбор языка и среды программирования) | 21.09.22 | 26.09.22 |
| Проектирование программного обеспечения.  (разработка структурной и функциональной схемы ПО, проектирование базы данных (инфологическое, ER-модель, физическая модель) | 26.09.22 | 25.10.22 |
| Разработка (программирование) и отладка программного продукта | 25.10.22 | 28.11.22 |
| Составление программной документации (написание руководства пользователя) | 28.11.22 | 06.12.22 |

## **Приложение Б – Листинг roles.php**

<?php  
require('db.php');  
class roles extends DB  
{  
 public function login($request)  
 {  
 $req = json\_decode($request);  
 $last\_name = $req->last\_name;  
 $name = $req->name;  
 $father\_name = $req->father\_name;  
 $password = $req->password;  
 $connect = $this->connect();  
 $sql = $connect->prepare('SELECT \* from users where last\_name=:last\_name and name=:name and father\_name=:father\_name and password=:pass');  
 $sql->execute([  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 'pass' => $password,  
 ]);  
 $data = $sql->fetch(PDO::*FETCH\_OBJ*);  
 if ($data) {  
 session\_start();  
 $\_SESSION['user'] = (object)[  
 'last\_name' => $data->last\_name,  
 'name' => $data->name,  
 'father\_name' => $data->father\_name,  
 'role' => $data->role  
 ];  
 }  
 }  
 public function registration($request)  
 {  
 $req = json\_decode($request);  
 $last\_name = $req->last\_name;  
 $name = $req->name;  
 $father\_name = $req->father\_name;  
 $password = $req->password;  
 $connect = $this->connect();  
 $sql = $connect->prepare('SELECT \* from users where last\_name=:last\_name and name=:name and father\_name=:father\_name and password=:pass');  
 $sql->execute(array(  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 "pass" => $password,  
 ));  
 $data = $sql->fetch(PDO::*FETCH\_OBJ*);  
 if ($data) {  
 return json\_encode([  
 'message' => "Такой пользователь существует"  
 ]);  
 }  
 $req = json\_decode($request);  
 $last\_name = $req->last\_name;  
 $name = $req->name;  
 $father\_name = $req->father\_name;  
 $connect = $this->connect();  
 $sql = $connect->prepare('SELECT \* from patients where last\_name=:last\_name and name=:name and father\_name=:father\_name');  
 $sql->execute(array(  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 ));  
 $data = $sql->fetch(PDO::*FETCH\_OBJ*);  
 if ($data) {  
 return json\_encode([  
 'message' => "Такой пользователь существует"  
 ]);  
 }  
 $sql = $connect->prepare("INSERT INTO users(last\_name,name,father\_name,password,role) values (:last\_name,:name,:father\_name,:pass,:role)");  
 $sql->execute([  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 "pass" => $password,  
 "role" => 3  
 ]);  
 $sql = $connect->prepare("INSERT INTO patients (last\_name,name,father\_name) values (:last\_name,:name,:father\_name)");  
 $sql->execute([  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 ]);  
 $sql = $connect->prepare('SELECT \* from users where last\_name=:last\_name and name=:name and father\_name=:father\_name and password=:pass');  
 $sql->execute([  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 "pass" => $password,  
 ]);  
 $data = $sql->fetch(PDO::*FETCH\_OBJ*);  
 if ($data) {  
 session\_start();  
 $\_SESSION['user'] = (object)[  
 'last\_name' => $data->last\_name,  
 'name' => $data->name,  
 'father\_name' => $data->father\_name,  
 'role' => $data->role  
 ];  
 return json\_encode([  
 'message' => 'Пользователь добавлен'  
 ]);  
 }  
 $sql = $connect->prepare('SELECT \* from patients where last\_name=:last\_name and name=:name and father\_name=:father\_name');  
 $sql->execute([  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 ]);  
 $data = $sql->fetch(PDO::*FETCH\_OBJ*);  
 if ($data) {  
 session\_start();  
 $\_SESSION['user'] = (object)[  
 'last\_name' => $data->last\_name,  
 'name' => $data->name,  
 'father\_name' => $data->father\_name,  
 ];  
 return json\_encode([  
 'message' => 'Пользователь добавлен'  
 ]);  
 }  
 }  
}