МИНОБРНАУКИ РОССИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**ВЫСШЕГО ОБРАЗОВАНИЯ**

**“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”**

**(ФГБОУ ВО «ВГУ»)**

Факультет компьютерных наук

Кафедра программирования и информационных технологий

Онлайн-система записи на прием к врачам с возможностью выбора специализации, врача и времени приема HealHub

Курсовая работа

по дисциплине

Технологии программирования

* + 1. Программная инженерия

Информационные системы и сетевые технологии

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.С. Тарасов, ст. преподаватель \_\_.\_\_.20\_\_

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.В.Черных, 3 курс, д/о

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.И.А.Мохамед, 3 курс, д/о

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_ К.А.М.Фахим, 3 курс, д/о

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.Д. Проскуряков, ассистент

Воронеж2024

Содержание

[Определения, обозначения и сокращения 3](#_Toc167917772)

[ВВЕДЕНИЕ 6](#_Toc167917773)

[1 Постановка задачи 7](#_Toc167917774)

[1.1 Функциональные требования к разрабатываемой системе 7](#_Toc167917775)

[1.1.1 Вход на веб-сайт 8](#_Toc167917776)

[1.1.2 Клиент 8](#_Toc167917777)

[1.1.3 Гость 9](#_Toc167917778)

[1.1.4 Врач 9](#_Toc167917779)

[1.1.5 Администратор 10](#_Toc167917780)

[1.2 Задачи, решаемые в процессе разработки 10](#_Toc167917781)

[1.3 Технические требования 11](#_Toc167917782)

[1.3.1 Языковые версии сайта 11](#_Toc167917783)

[1.3.2 Группы пользователей 11](#_Toc167917784)

[1.3.3 Функциональность сайта 12](#_Toc167917785)

[1.4 Требования к интерфейсу 12](#_Toc167917786)

[2 Анализ предметной области 13](#_Toc167917787)

[2.1 Обзор аналогов 13](#_Toc167917788)

[3 Реализация 14](#_Toc167917789)

[3.2 Реализация серверной части веб-приложения 14](#_Toc167917790)

[3.3 Реализация клиентской части веб-приложения 15](#_Toc167917791)

[ЗАКЛЮЧЕНИЕ 17](#_Toc167917792)

[Список использованных источников 18](#_Toc167917793)

[ПРИЛОЖЕНИЕ А 19](#_Toc167917794)

[Приложение Б 20](#_Toc167917795)

[ПРИЛОЖЕНИЕ В 22](#_Toc167917796)

[Приложение Г 23](#_Toc167917797)

[Приложение Д 24](#_Toc167917798)

# Определения, обозначения и сокращения

**Техническое Задание (ТЗ)** – документ, содержащий описание требований и условий, необходимых для выполнения конкретной задачи или проекта.

**Неавторизованный Пользователь (Гость)** – пользователь, не прошедший процедуру авторизации на платформе и имеющий ограниченный доступ к функционалу системы.

**Администратор** – пользователь с расширенными правами доступа, ответственный за управление и настройку системы, включая управление пользователями и техническую поддержку.

**Платформа** – совокупность аппаратного и программного обеспечения, обеспечивающая выполнение и поддержку работы приложений или услуг.

**Интерактивная карта** – карта города, предоставляющая пользователям выбирать определенного врача на основании его местоположения.

**Уведомления** – сообщения, отправляемые пользователям для информирования о событиях, изменениях и/или других важных действиях в системе.

**Авторизация** – процесс проверки подлинности пользователя с целью предоставления доступа к определенным частям и/или функционалу веб-сайта.

**Онлайн-запись** – функция, позволяющая пользователям регистрироваться на определенные мероприятия, услуги или консультации через интернет.

**Регистрация** – процесс создания новой учетной записи пользователя в системе, включающий ввод личной информации и настройку учетных данных.

**Оптимизация** – процесс улучшения производительности, эффективности и скорости работы системы или ее компонентов.

**ID** – уникальный идентификатор, используемый для однозначного обозначения объектов, пользователей или записей в системе.

**Статистика** – совокупность данных, собранных и проанализированных для получения информации о работе системы, пользовательской активности или других аспектах.

**Email** – электронная почта, используемая для отправки и получения сообщений через интернет.

**База Данных (БД)** – организованная структура для хранения, управления и извлечения данных.

**Верстка** – процесс создания структуры и внешнего вида веб-страницы с использованием HTML, CSS и иногда JavaScript.

**Макет сайта** – графическое представление структуры и дизайна веб-страницы, используемое как шаблон для её создания.

**REST API** – интерфейс программирования приложений, основанный на принципах REST (Representational State Transfer), позволяющий взаимодействовать с веб-сервисами через стандартизированные HTTP-запросы.

**Front-end** – клиентская часть веб-приложения, отвечающая за отображение интерфейса и взаимодействие с пользователем, обычно реализованная с помощью HTML, CSS и JavaScript.

**Back-end** – серверная часть веб-приложения, обрабатывающая данные, запросы и бизнес-логику, обычно реализованная с использованием серверных языков программирования и баз данных.

**Сервер** – компьютер или программное обеспечение, предоставляющее сервисы и ресурсы другим компьютерам или программам в сети.

**Java** – универсальный объектно-ориентированный язык программирования, широко используемый для создания различных типов программного обеспечения, веб-приложений и мобильных приложений.

**Spring** – фреймворк для разработки Java-приложений, облегчающий создание корпоративных и веб-приложений с помощью инверсии управления и внедрения зависимостей.

**MySQL** – система управления реляционными базами данных (СУБД), широко используемая для хранения данных в веб- и других приложениях и управления ими.

**HTML** – язык разметки, используемый для создания структуры веб-страниц с помощью различных элементов и тегов.

**CSS** – язык таблиц стилей, который используется для оформления веб-страниц, задавая внешний вид и расположение элементов на странице.

**JavaScript** – интерпретируемый язык программирования, используемый для добавления интерактивности на веб-страницы. Включает в себя обработку событий, динамическое обновление содержимого и пр.

**React** – JavaScript-библиотека для создания пользовательских интерфейсов, разработанная компанией Facebook. Упрощает создание масштабируемых и интерактивных веб-приложений.

**Конверсия** – показывает процент конверсии, который обычно указывает на успешность выполнения определенного действия по отношению к визитам.

**Достижения цели** – указывает количество раз, когда была достигнута конкретная цель, например, отправка формы или завершение регистрации.

**Целевые визиты** – показывает количество визитов, во время которых была выполнена конкретная цель.

ВВЕДЕНИЕ

Современная медицина постоянно стремится к совершенствованию и инновациям, с целью обеспечения населению более доступного, качественного и эффективного медицинского обслуживания. Учитывая это, важное значение имеет возможность онлайн-записи на прием к врачам, позволяющая выбрать специалиста, конкретное время и удобное для вас местоположение.

Одной из таких инновационных платформ является система HealHub. Это онлайн-приложение, которое облегчает и ускоряет процесс записи на прием к врачам, предоставляя пользователям широкий выбор специалистов различных медицинских направлений.

Основной целью нашего проекта является обеспечение максимального удобства, доступности и эффективности медицинского обслуживания для всех граждан, кто в этом нуждается. Мы уверены, что использование онлайн-системы записи на прием в качестве ключевого инструмента поможет пациентам значительно сократить время ожидания и выбрать наиболее подходящее время и специалиста для решения именно их медицинской проблемы.

За счет анализа данных исследований в области здравоохранения и активного учета обратной связи от пользователей, мы стремимся постоянно совершенствовать, улучшать и расширять функционал приложения HealHub, чтобы обеспечить максимальное удобство и эффективность его использования для каждого нашего пользователя.

Мы убеждены, что использование системы HealHub в жизнь людей поможет оптимизировать и упростить процесс записи на прием, повысить общий уровень обслуживания пациентов и значительно улучшить качество медицинской помощи в целом.

# 1 Постановка задачи

Цель данного проекта заключается в разработке современного веб-приложения, которое способствует улучшению доступа к медицинским услугам посредством автоматизации системы записи на прием к врачу. Это приложение ориентировано на создание веб-приложения, позволяющему пациентам самостоятельно выбирать врача нужной специализации и записываться на прием через интерактивную карту. Пациенты смогут выбирать удобное время для визита, а также получать подтверждения и напоминания о записи на электронную почту.

Кроме того, проект предусматривает разработку интерфейса, который обеспечит пользователям легкость и простоту в использовании всех функций приложения. Интерактивная карта будет включать в себя актуальную информацию о расположении медицинских учреждений, доступных врачах, а также их расписаниях, что позволит пациентам быстро находить и выбирать наиболее подходящие варианты.

Система уведомлений будет интегрирована с электронной почтой пациента, что позволит своевременно получать напоминания о предстоящих визитах, а также информацию об изменениях в расписании или отмене записи.

Таким образом, основной задачей проекта является создание эффективного инструмента, который упростит процесс записи на прием к врачу, сделает его более доступным и удобным для широкого круга пользователей, а также поможет медицинским учреждениям оптимизировать работу с пациентами.

# 1.1 Функциональные требования к разрабатываемой системе

* Разработка механизма регистрации и авторизации пользователей;
* Создание интерактивной карты для выбора врача с учетом их местоположения;
* Реализация системы напоминаний о предстоящих приемах;
* Обеспечение возможности получения оповещений об изменениях в записи приема;
* Предоставление пользователю возможности связи с администратором.

Пользователь будет иметь одну из ролей:

* Неавторизованный пользователь;
* Администратор;
* Клиент;
* Врач.

# 1.1.1 Вход на веб-сайт

* При первом заходе на сайт пользователь видит основную страницу с приветственным текстом;
* В правом верхнем углу пользователь может зарегистрироваться, нажав на кнопку «Регистрация»;
* В правом верхнем углу пользователь может войти в свой профиль, нажав на кнопку «Войти», если до этого он уже регистрировался на сайте;
* В правом верхнем углу пользователь может узнать информацию о сайте HealHub, нажав на кнопку «О нас»;
* В левом верхнем углу пользователь, нажав на логотип «HealHub», может попасть на основную страницу сайта;

# 1.1.2 Клиент

Авторизированный пользователь может:

* Просмотреть профиль врача;
* Просмотреть данные своего аккаунта;
* Получить оповещение о записи на почту;
* Просмотреть информацию о текущих записях;
* Восстановить забытый пароль;
* Просмотреть детали приема;
* Просмотреть истории приемов;
* Просмотреть информацию о компании HealHub;
* Сделать выбор определенного времени записи к врачу, если у специалиста есть свободное время;
* Выбрать нужного врача на карте;
* Выбрать врача нужной врачебной специализации;
* При необходимости иметь возможность отмены приема у выбранного им специалиста;
* Возможности выбора врачей по регионам и городам Российской Федерации, если в требуемом населенном пункте имеются нужные ему специалисты;

# 1.1.3 Гость

Гость (Неавторизированный пользователь) может:

* Просмотреть профиль врача;
* Выбрать врача нужной специализации;
* Возможность зарегистрироваться на сайте для получения возможности записаться к желаемому врачу;
* Возможности войти в свой профиль, если пользователь уже был зарегистрирован на сайте.

# 1.1.4 Врач

Врач может:

* Просмотреть историю записей клиента;
* Просмотреть свою учетную запись;
* При необходимости может отменить прием определенного пациента;
* Просмотреть информацию о приеме;
* При необходимости может редактировать детали приема определенного пациента;
* Просмотреть расписание приемов.

# 1.1.5 Администратор

Администратор может:

* Сбросить пароль пользователя;
* Удалить учетную запись пользователя;
* Просмотреть данные пользователя;
* Редактировать данные пользователя;
* Управлять специализациями врачей;
* Просматривать статистику сайта (посещаемость сайта (за день, неделю, месяц, год и все время), записей пациентов) в виде графиков и диаграмм;
* Найти определенного пользователя по уникальному id, ФИО, email или номеру телефона;
* Просматривать записи о приемах.

# 1.2 Задачи, решаемые в процессе разработки

Перед проектом были поставлены следующие задачи:

* Написание технического задания;
* Описание разрабатываемой системы UML диаграммами;
* Реализация баз данных;
* Реализация ролей (Неавторизированный пользователь, Администратор, Клиент, Врач);
* Реализация функциональных возможностей ролей;
* Реализация функциональных возможностей веб-сайта, описанных в ТЗ проекта;
* Создание макета дизайна сайта и его реализация;
* Верстка макета сайта;
* Проведение тестирования;
* Написание курсовой работы по проделанной работе;

# 1.3 Технические требования

# 1.3.1 Языковые версии сайта

Все страницы веб-приложения спроектированы и реализованы с учетом поддержки русского языка в качестве основного языка интерфейса приложения.

Это включает в себя не только текстовое содержание, но и все элементы пользовательского интерфейса, к примеру такие как кнопки. Гарантируется, что все пользователи, просматривающие и использующие приложение, будут иметь доступ к информации на русском языке.

# 1.3.2 Группы пользователей

Веб-приложение разработано с учетом многоуровневой системы доступа, предоставляя возможности для четырех различных категорий пользователей: Гостя, Клиента, Врача и Администратора. Эта структура пользовательских ролей гарантирует, что каждый тип пользователей будет иметь свои уникальные возможности и функциональности в приложении, соответствующие их потребностям и ролям в системе. Такой подход обеспечивает оптимальное управление доступом к данным и функциям, повышая безопасность и эффективность использования приложения для всех его пользователей.

# 1.3.3 Функциональность сайта

Веб-приложение должно обеспечивает полный спектр функциональных возможностей для пользователей, включая авторизацию, просмотр подробной информации в профиле врача с полной информацией о специализации, опыте и образовании, возможность выбора врача в соответствии с конкретной медицинской областью, а также предоставление личного кабинета с персонализированным доступом для всех категорий пользователей. Это включает в себя возможность для гостей регистрации и получения базовых возможностей, для клиентов — управление своими записями и профилем, для врачей — управление своим расписанием, записями и своим профилем, а для администраторов — управление данными и статистикой приложения.

# 1.4 Требования к интерфейсу

Все страницы веб-приложения оформлены в одном стиле с целью создания единого и последовательного визуального образа для пользователей. Это включает в себя согласованное использование цветовой гаммы, шрифтов, элементов дизайна и компоновки контента на всех страницах. Каждый элемент интерфейса видим и различим на фоне других элементов, что обеспечивает удобство использования и навигации для пользователей. Основное меню сайта располагается в верхней части экрана на всех страницах, обеспечивая быстрый доступ к основным функциям и разделам приложения для пользователей.

# 2 Анализ предметной области

# 2.1 Обзор аналогов

# 3 Реализация

Страницы сайта корректно отображаются в браузерах Google Chrome версии 122.0.6261.111 и выше, Microsoft Edge версии 121.0.2277.83 и выше, Yander Browser (Яндекс Браузер) версии 23.11.3.955 и выше.

Веб-приложение соответствует шаблону клиент-серверного приложения и разделяется на back-end и front-end с помощью средств REST API.

# 3.2 Реализация серверной части веб-приложения

Веб-приложение HealHub реализовано по архитектуре клиент-сервер с использованием REST API для обеспечения коммуникации между клиентской и серверной частями. Серверная часть отвечает за основные бизнес-процессы, такие как управление пользователями, записи на прием к врачам, управление клиниками и специальностями врачей.

Основные требования к серверной части включают:

* Регистрация и авторизация пользователей;
* Управление профилями пользователей и врачей;
* Организация расписаний врачей и управление записями на прием;
* Обработка запросов на предоставление информации о клиниках и их услугах.

Для реализации программно-аппаратной части были выбраны следующие технологии:

* Язык программирования Java 17;
* Фреймворк Spring Boot 2.6.3 для создания RESTful веб-сервисов;
* База данных MySQL для хранения всех данных приложения;
* Spring Security и JWT для аутентификации и авторизации пользователей.

Все данные обрабатываются с помощью контроллеров Spring MVC, которые принимают HTTP-запросы и взаимодействуют с сервисным слоем для выполнения бизнес-логики. Данные модели передаются через слой Data Transfer Objects (DTO), обеспечивая изоляцию доменной модели от клиентских запросов.

# 3.3 Реализация клиентской части веб-приложения

Клиентская часть HealHub разрабатывается с использованием современных технологий фронтенд-разработки для обеспечения интерактивности и удобства использования веб-приложения.

Для реализации клиентской части были выбраны следующие технологии:

* Язык программирования JavaScript;
* Библиотека React 18.2.0 для построения пользовательского интерфейса;
* CSS для стилизации компонентов;
* HTML для структуры веб-страниц.

Клиентская часть структурирована в виде SPA (Single Page Application), что позволяет пользователям взаимодействовать с приложением без перезагрузки страницы.

React Router используется для организации навигации между компонентами.

Redux применяется для управления состоянием приложения на клиентской стороне, что упрощает управление состоянием и делает поведение приложения более предсказуемым.

Интерфейс разделен на страницы: главная, профиль пользователя, страницы управления записями на прием, страницы клиник и врачей, а также административный интерфейс для управления пользователями и контентом. Вся клиентская логика тщательно разработана для обеспечения максимальной отзывчивости и удобства пользователя.

Все компоненты и страницы разработаны с учетом мобильной адаптации, обеспечивая корректное отображение и функциональность на различных устройствах и платформах

ЗАКЛЮЧЕНИЕ

В ходе выполнения данного курсового проекта была разработана современная и функциональная веб-платформа HealHub, предназначенная для автоматизации процесса записи на прием к врачу. Применение Java 17 и Spring Boot 2.6.3 для реализации серверной части, а также использование React 18.2.0 для разработки клиентской части, позволило создать надежное и высокопроизводительное приложение, соответствующее современным стандартам разработки программного обеспечения.

HealHub предоставляет пользователям удобный интерфейс для выбора специалистов и записи на прием, интегрированный с интерактивной картой и системой уведомлений. Разработанная система обеспечивает возможность для пользователей просматривать профили врачей, управлять своими записями и получать напоминания о предстоящих визитах, что значительно повышает уровень доступности и удобства медицинского обслуживания.

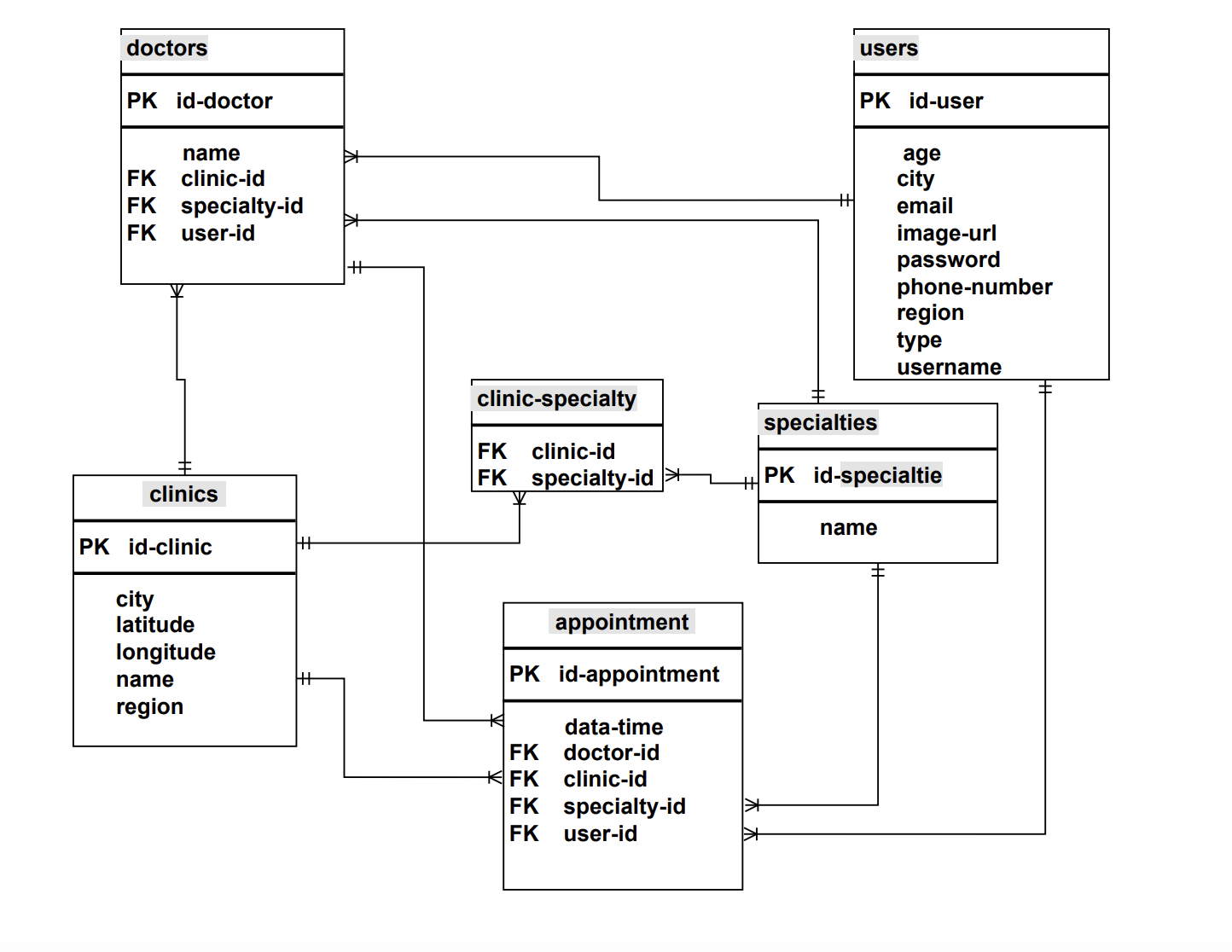
В ходе разработки были выполнены все поставленные задачи, включая создание функциональных возможностей для различных ролей пользователей, разработку базы данных, реализацию бизнес-логики.

HealHub способствует оптимизации работы медицинских учреждений и повышению эффективности взаимодействия с пациентами, что в конечном итоге положительно сказывается на общем уровне медицинского обслуживания. Полученные результаты и опыт могут быть использованы для дальнейшего совершенствования и расширения функциональности системы, что сделает ее еще более полезной и востребованной для широкого круга пользователей.

Список использованных источников

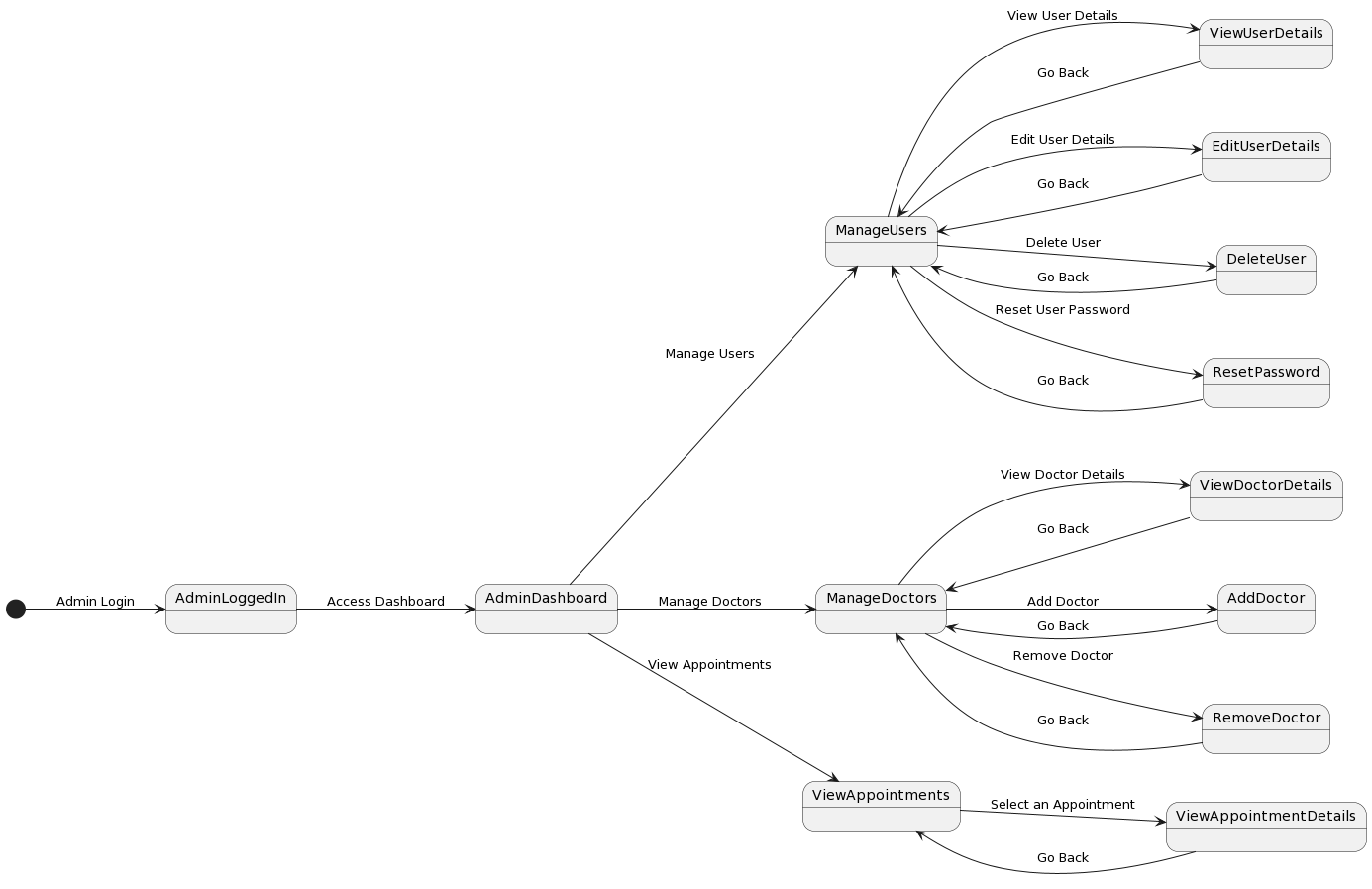
1. Тут будут использованные источники [Электронный ресурс]. – Режим доступа: URL: –

ПРИЛОЖЕНИЕ А

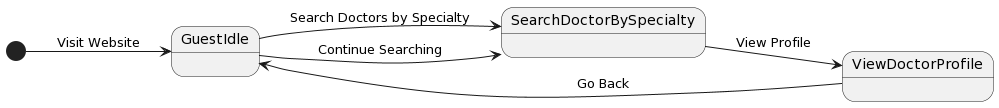


1. ER диаграмма

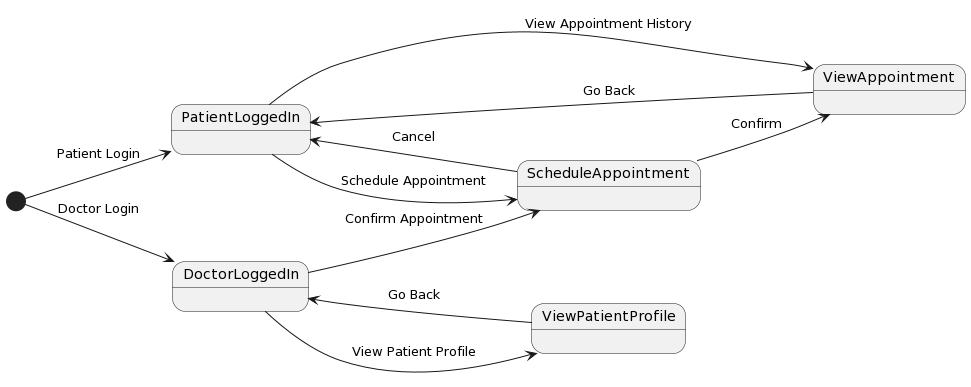
Приложение Б



1. Диаграмма состояний администратора

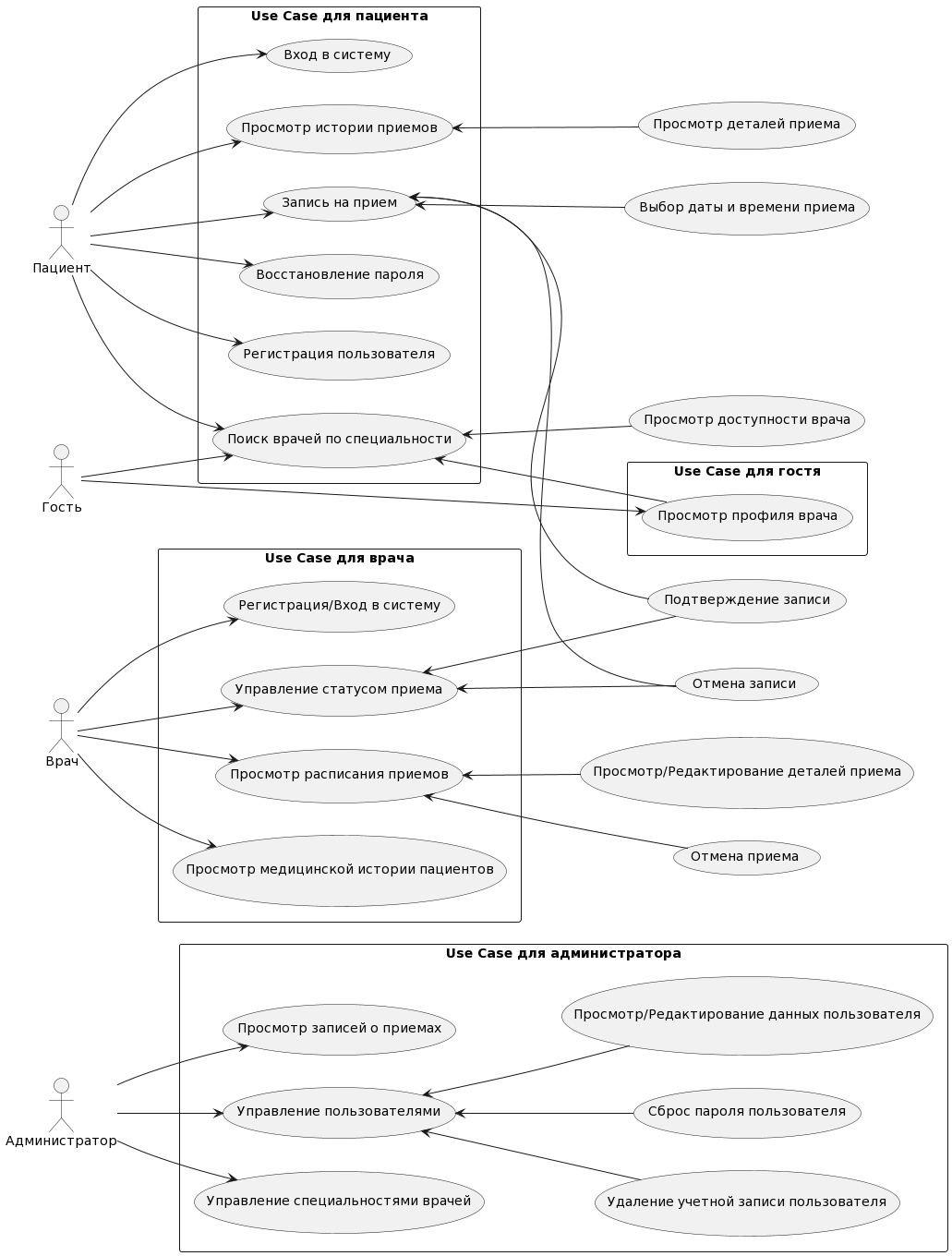


1. Диаграмма состояний гостя

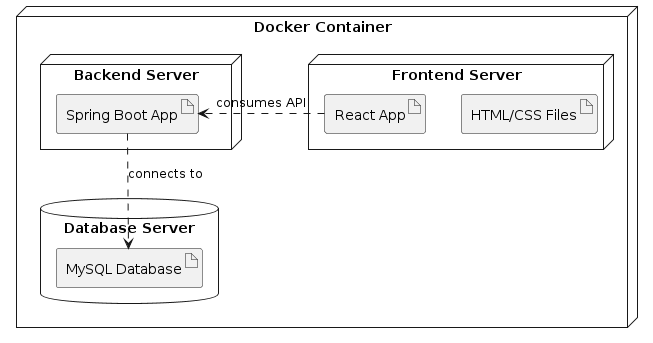


1. Диаграмма состояний пациент и врач

ПРИЛОЖЕНИЕ В

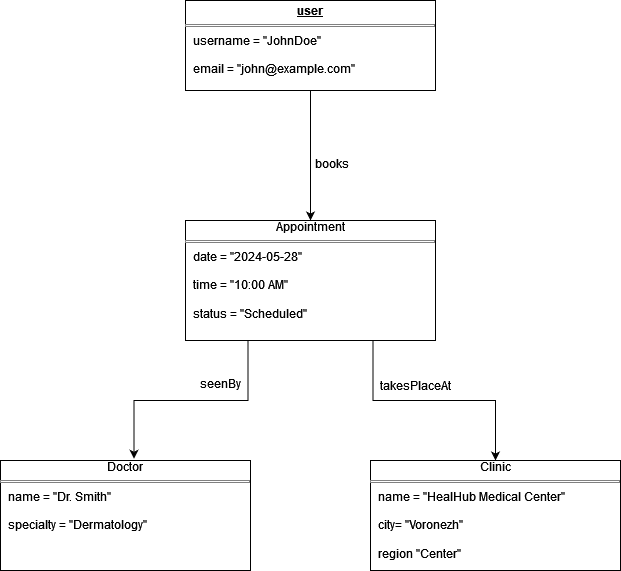


1. Use-Case диаграмма

Приложение Г  


1. Deployment diagram

Приложение Д



1. Object Diagram