МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой, д.ф.-м.н.,
профессор

_____ Л.Б. Соколинский

«___»_____ 2024 г.

Разработка веб-сайта медицинского центра ЮУрГУ

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 09.03.04.2024.308-335.ВКР

Научный руководитель,
доцент кафедры СП, к.ф.-м.н.
_____ Т.Ю. Маковецкая

Автор работы,
студент группы КЭ-403
_____ А.М. Галлямов

Ученый секретарь
(нормоконтролер)
_____ И.Д. Володченко
«___»_____ 2024 г.

Челябинск, 2024 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
29.01.2024 г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы бакалавра
студенту группы КЭ-403
Галлямову Артуру Марселевичу,
обучающемуся по направлению
09.03.04 «Программная инженерия»

1. **Тема работы** (утверждена приказом ректора от 22.04.2024 г. № 764-13/12)

Разработка веб-сайта медицинского центра ЮУрГУ.

2. **Срок сдачи студентом законченной работы:** 03.06.2024 г.

3. **Исходные данные к работе**

3.1. Документация Laravel. [Электронный ресурс] URL:
https://laravel.com/docs/10.x (дата обращения: 01.02.2024 г.).

3.2. Дронов В.А. Laravel: быстрая разработка динамических
Web-сайтов на PHP, MySQL, HT ML и CSS. // СПб.: БХВ-Петербург,
2018 – 768 с.

3.3. Бейли Л. Изучаем PHP и MySQL. / Л. Бейли, М. Моррисон // Москва:
Эксмо, 2010 – 800 с.

3.4. Современный учебник JavaScript [Электронный ресурс] URL:
https://learn.javascript.ru (дата обращения: 01.02.2024 г.).

4. **Перечень подлежащих разработке вопросов**

4.1. Провести анализ аналогичных проектов.

4.2. Спроектировать и реализовать базу данных веб-сайта.

4.3. Спроектировать и реализовать веб-сайт.

4.4. Провести тестирование веб-сайта.

5. Дата выдачи задания: 29.01.2024 г.

Научный руководитель,
доцент кафедры СП, к.ф.-м.н. Т.Ю. Маковецкая

Задание принял к исполнению А.М. Галлямов

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность

В современном мире Интернет стал неотъемлемой частью жизни людей. Он позволяет получать информацию, развлекаться, общаться и решать повседневные проблемы, возникающие в жизни. Одной из таких проблем является выбор качественного медицинского обслуживания.

Для медицинских организаций веб-сайт является эффективным инструментом привлечения и удержания клиентов, повышения доверия и лояльности, а также поддержания обратной связи. Кроме того, веб-сайт может помочь пользователю и будущему пациенту максимально удобно и быстро получить информацию о контактах клиники, требуемых услугах, сотрудниках, которые их предоставляют, и оставить свой отзыв об опыте пребывания в клинике.

В современном мире, где интернет стал основным источником информации для большинства людей, наличие хорошо оптимизированного веб-сайта является ключевым фактором успеха для любой медицинской организации. Продвижение веб-сайта становится не просто элементом престижа, но и необходимым условием для привлечения новых пациентов, установления доверительных отношений с существующей клиентурой и создания позитивного имиджа организации в интернете.

Постановка задачи

Целью выпускной квалификационной работы является разработка веб-сайта медицинского центра ЮУрГУ. Для достижения поставленной цели необходимо решить следующие задачи:

1) анализ аналогичных проектов;
2) проектирование и реализация базы данных веб-сайта;
3) проектирование и реализация веб-сайта;
4) тестирование веб-сайта.

Структура и содержание работы

Работа состоит из введения, четырех глав, заключения, списка литературы и приложения. Объем работы составляет 45 страниц, объем списка литературы – 16 источников.

В первой главе описывается анализ предметной области работы, что включает в себя обзор аналогичных или схожих проектов и анализ функциональных и нефункциональных требований к разрабатываемому приложению. Помимо этого, в данной главе представлена диаграмма вариантов использования системы.

Вторая глава посвящена проектированию системы. Также данная глава включает в себя описание архитектуры и схему базы данных системы, диаграмму деятельности одного из вариантов использования системы и проектирование интерфейса в виде макетов некоторых страниц разрабатываемого веб-сайта.

В третьей главе представлены элементы реализации компонентов проекта, что включает в себя описание реализации архитектуры веб-приложения, листинги разработанного веб-приложения и список выбранных инструментов для его реализации. Помимо этого, к ним приведены скриншоты основных интерфейсов веб-приложения.

В четвертой главе представлен набор тестов функционального тестирования веб-сайта и результаты его прохождения.

В приложении содержатся спецификации ключевых вариантов использования веб-приложения.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

В данной главе представлен анализ предметной области, который включает в себя описание предметной области, анализ аналогичных или схожих проектов и анализ требований к системе.

1.1. Описание предметной области

Медицинский центр предлагает широкий спектр услуг, которые охватывают различные области медицины. Эти услуги оказываются квалифицированными врачами, каждый из которых специализируется в определенной области и работает по индивидуальному графику. Один сотрудник может занимать несколько должностей, что расширяет его компетенцию и позволяет оказывать разнообразные услуги. Вакансии регулярно публикуются, чтобы информировать потенциальных сотрудников о наличии свободных мест.

Все услуги, предоставляемые медицинским центром, распределены по разным направлениям или категориям. Каждая услуга имеет свою стоимость. При этом стоит отметить, что цены могут быть снижены в рамках акций, что делает услуги более доступными для пациентов. Акции имеют конкретные даты начала и окончания, которые обозначают сроки их действия.

Медицинский центр также активно информирует пациентов о своей деятельности, публикуя новости из жизни центра. Это может включать в себя информацию о новых услугах, изменениях в графике работы, достижениях персонала и других важных событиях.

Кроме того, медицинский центр предоставляет доступ к ряду важных документов, таких как справки, политики и регламенты. Эти документы важны для обеспечения прозрачности и понимания пациентами своих прав и обязанностей при получении медицинских услуг.

Помимо всего прочего, медицинский центр ЮУрГУ расположен по двум адресам и предлагает услуги по разным направлениям в каждом.

1.2. Анализ аналогичных проектов

Для представления разрабатываемого проекта был выполнен анализ аналогичных или схожих проектов. Было найдено несколько веб-сайтов, представляющих медицинские организации.

Клиника «Источник»

Клиника «Источник» [1] – современная многопрофильная клиника, которая оказывает качественную медицинскую помощь более чем по 40 направлениям.

Веб-сайт клиники «Источник» позволяет получить общую информацию о клинике, просмотреть список всех сотрудников организации, разузнать о предлагаемых услугах и акциях. Помимо этого, так как организацию представляют несколько клиник, расположенных по разным адресам, сайт предоставляет информацию по их местоположению, времени работы и контактам. Местоположение клиники показывается на встроенной в страницу карте. Главная страница веб-сайта клиники «Источник» представлена на рисунке 1.

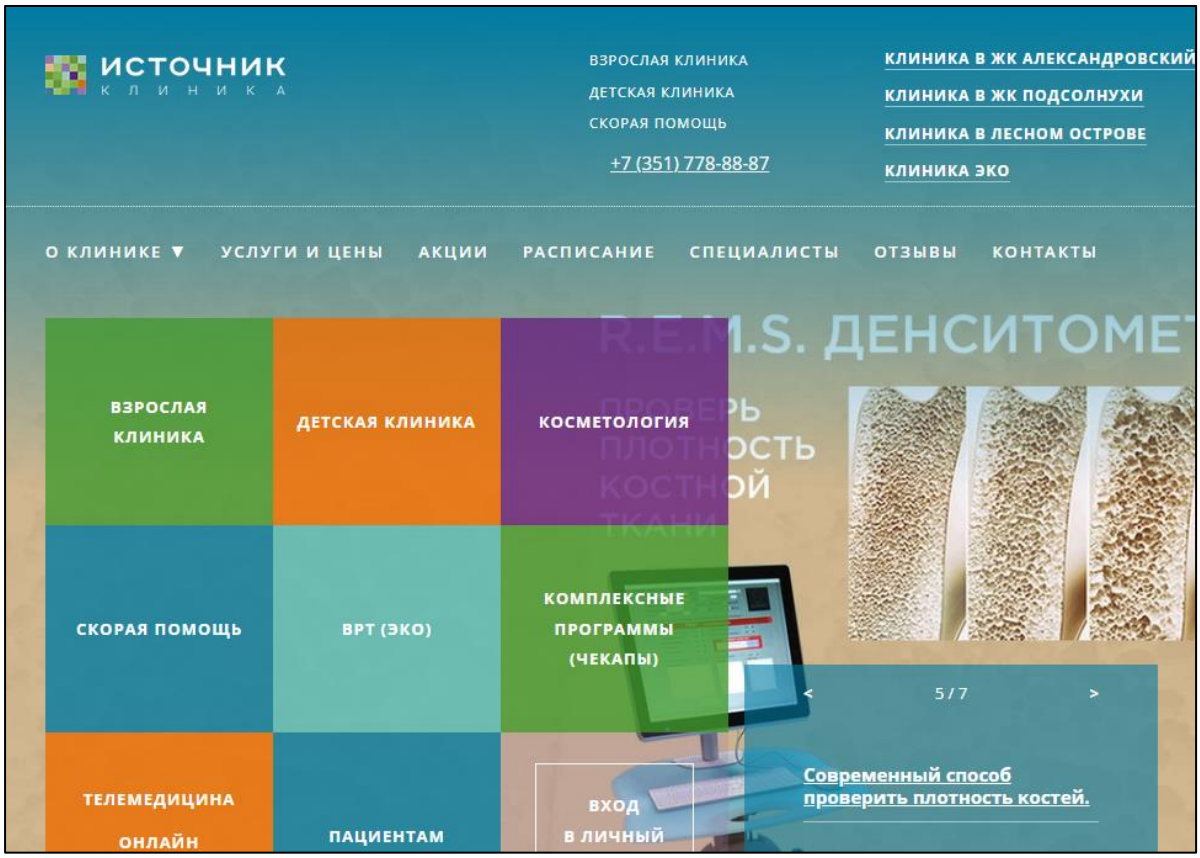

Рисунок 1 – Скриншот главной страницы клиники «Источник»

Данный веб-сайт имеет личный кабинет, позволяющий пользователям создать учетную запись и авторизоваться на сайте. Каждый пользователь может оставить отзыв о конкретном специалисте, дав ему оценку от 1 до 5 и оставив небольшое сообщение.

Веб-сайт предоставляет подробную информацию о сотрудниках клиники, включая их стаж должности, стаж, образование, направления деятельности. Для просмотра также предоставлены фотографии сотрудников.

Предоставляемые услуги разбиты по клиникам, где они оказываются, и направлениям. Веб-сайт позволяет пользователю записаться на прием, оставив свой номер, адрес электронной почты и имя.

Медицинский центр «Медеор»

Медицинский центр «Медеор» [2] – многопрофильная клиника, предлагающая полный комплекс медицинских услуг.

Как и веб-сайт клиники «Источник», веб-сайт клиники «Медеор» позволяет пользователю быстро получить необходимую информацию о медицинском центре. Ссылки на категории услуг, предоставляемых клиникой, уже встроены в навигационную панель сайта, там же есть ссылка на страницу сотрудников и страницу цен. На главной странице сайта также представлены новости центра, ссылки на услуги с символьным обозначением каждой, встроенный видеоролик, рассказывающий о клинике, акции и фотографии центра. Также присутствует строка поиска по всему сайту, что позволяет быстро найти необходимую информацию. По кнопке, удобно расположенной в нижнем правом углу экрана, можно найти ссылки на социальные сети, в которых представлена клиника. Помимо всего прочего, сайт имеет версию для слабовидящих.

Информация о сотрудниках, доступная пользователю включает в себя стаж, образование, сведения об участии в конференциях, награды, работы, специализацию и отзывы со сторонних источников, если такие есть.

В отличии от веб-сайта клиники «Источник», данный сайт не предоставляет возможности пользователю оставлять отзывы о своем опыте пребывания в медицинском центре непосредственно через сайт.

«Медеор» также имеет свое мобильное приложение, ссылка на которое доступна на главной странице веб-сайта.

Главная страница веб-сайта медицинского центра «Медеор» представлена на рисунке 2.

Рисунок 2 – Скриншот главной страницы медицинского центра «Медеор»

Медицинский центр «Лотос»

Медицинский центр «Лотос» [3] – многопрофильное негосударственное лечебно-профилактическое учреждение на Южном Урале.

По аналогии с другими рассмотренными примерами, веб-сайт клиники «Лотос» предоставляет информацию об услугах, ценах, специалистах и контактах через навигационную панель. На главной странице веб-сайта

показаны достижения и статистика медицинского центра. Помимо этого, веб-сайт предоставляет возможность вызвать врача на дом и записи на прием онлайн. Присутствует поиск по сайту, отправка отзывов о клинике непосредственно через сайт, версия сайта для слабовидящих. Адреса филиалов клиники показаны при помощи интерактивной карты.

Главная страница веб-сайта медицинского центра «Лотос» представлена на рисунке 3.

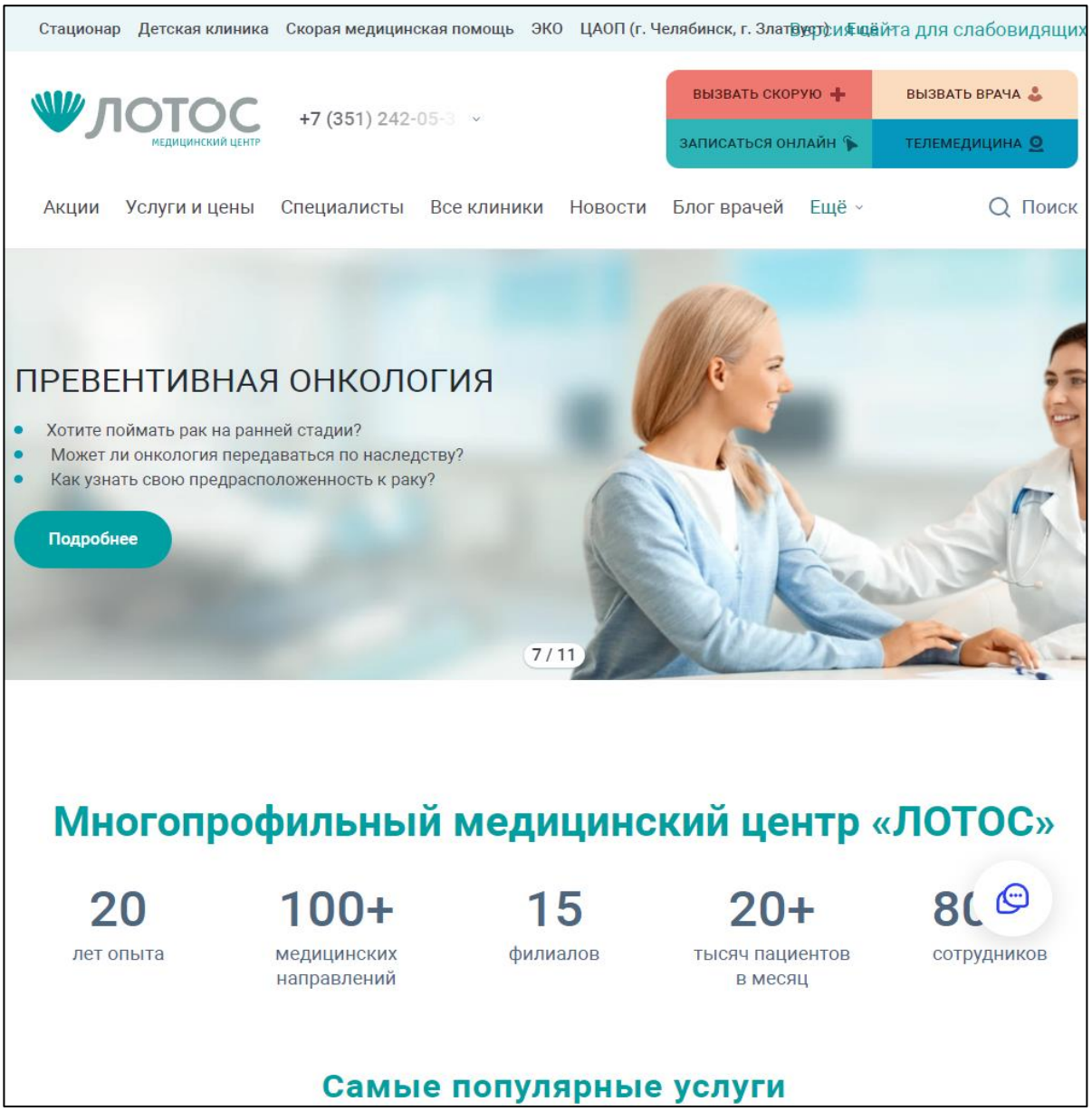

Рисунок 3– Скриншот главной страницы медицинского центра «Лотос»

Отличительной особенностью веб-сайта на фоне остальных примеров является блог врачей. В данном разделе сотрудники медицинского

центра делятся своими статьями по медицинским направлениям, которые написаны понятным языком и могут быть полезны пациентам. Статьи распределены по темам направлений медицины и имеют функцию сортировки. Страница блога врачей представлена на рисунке 4.

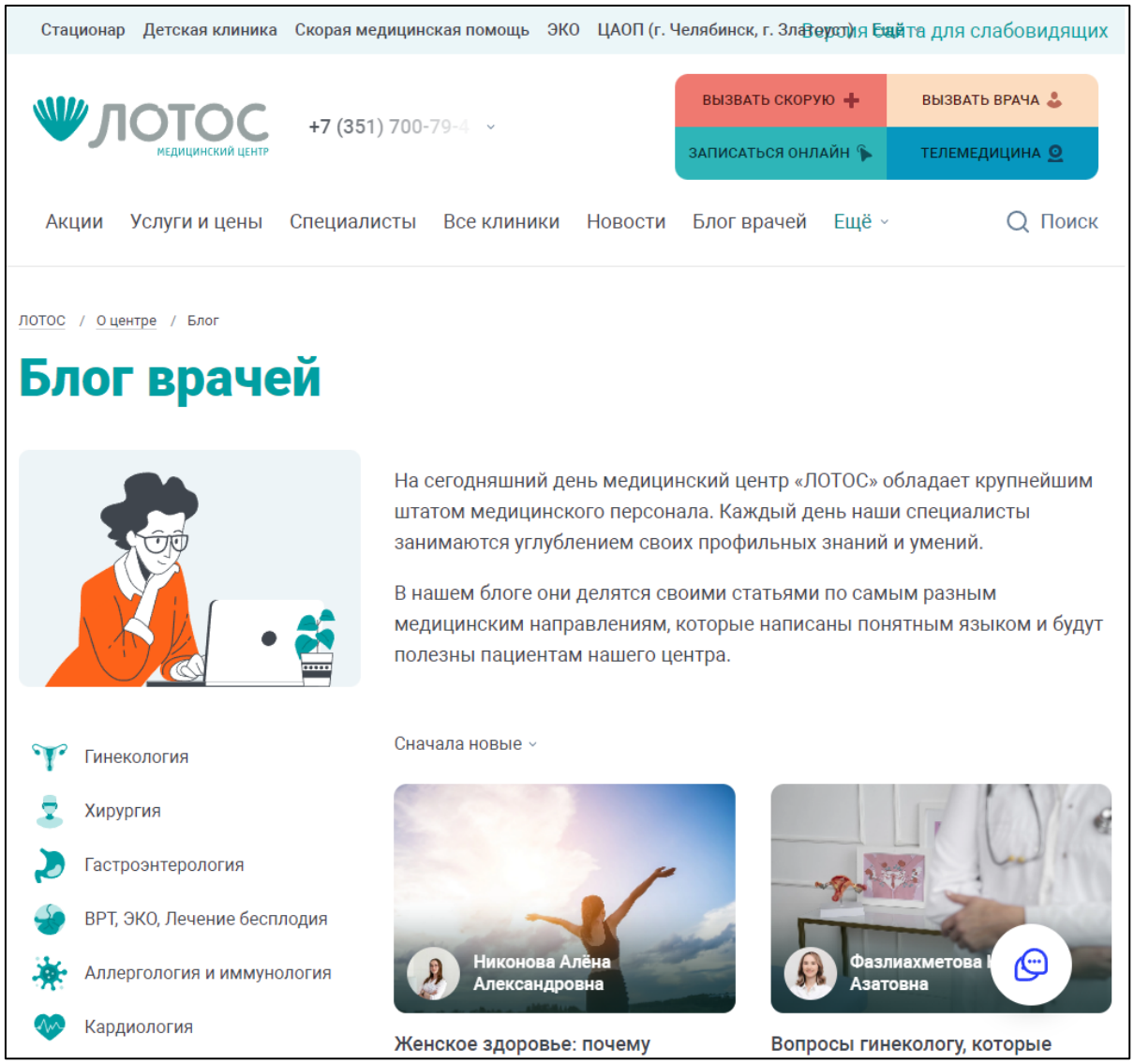

Рисунок 4– Скриншот блога врачей медицинского центра «Лотос»

1.3. Анализ требований к системе

Проведя анализ аналогичных проектов, можно выделить ряд функциональных и нефункциональных требований к проектируемой системе.

Функциональные требования

Разрабатываемый веб-сайт должен удовлетворять следующим функциональным требованиям:

Веб-приложение должно предоставлять:

1) возможность просмотра общей информации о клинике, включая контактную информацию, адреса отделений, ссылки на лицензии;

2) возможность просмотра информации о специалистах клиники;

3) возможность просмотра информации об услугах, оказываемых в клинике, ценах на них и акциях;

4) возможность просмотра отзывов о клинике и возможность отправлять их;

5) возможность просматривать документы, касающиеся клиники: политики, лицензии, справки;

6) функционал для модерации отзывов администратором;

7) функционал для редактирования контента сайта, включая новости, документы, информацию о сотрудниках, услугах, ценах, акциях.

Нефункциональные требования

Разрабатываемое приложение должно удовлетворять следующим нефункциональным требованиям:

1) веб-приложение должно поддерживаться в последних версиях всех популярных браузеров;

2) пользовательский интерфейс веб-сайта должен быть адаптивным: поддерживать разрешения экрана всех современных устройств;

3) контент веб-приложения должен соответствовать приказу Минздрава РФ от 30.12.2014 г. N 956Н [4].

Диаграмма вариантов использования

Согласно сформулированным требованиям к системе, была построена диаграмма вариантов использования приложения на языке графического описания для объектного моделирования UML. UML (Unified Modelling Language) [5] содержит стандартный набор диаграмм и нотаций самых разнообразных видов для документирования, определения и проектирования программных систем.

В проектируемой системе можно выделить 2 основных актера.

1. Пользователь – рядовой пользователь сети Интернет, взаимодействующий с веб-сайтом.

2. Администратор – сотрудник медицинского центра, обладающий возможностью управления контентом веб-сайта и модерации отзывов.

Спецификации основных вариантов использования проектируемой системы представлены в приложении.

На рисунке 5 представлена диаграмма вариантов использования системы.

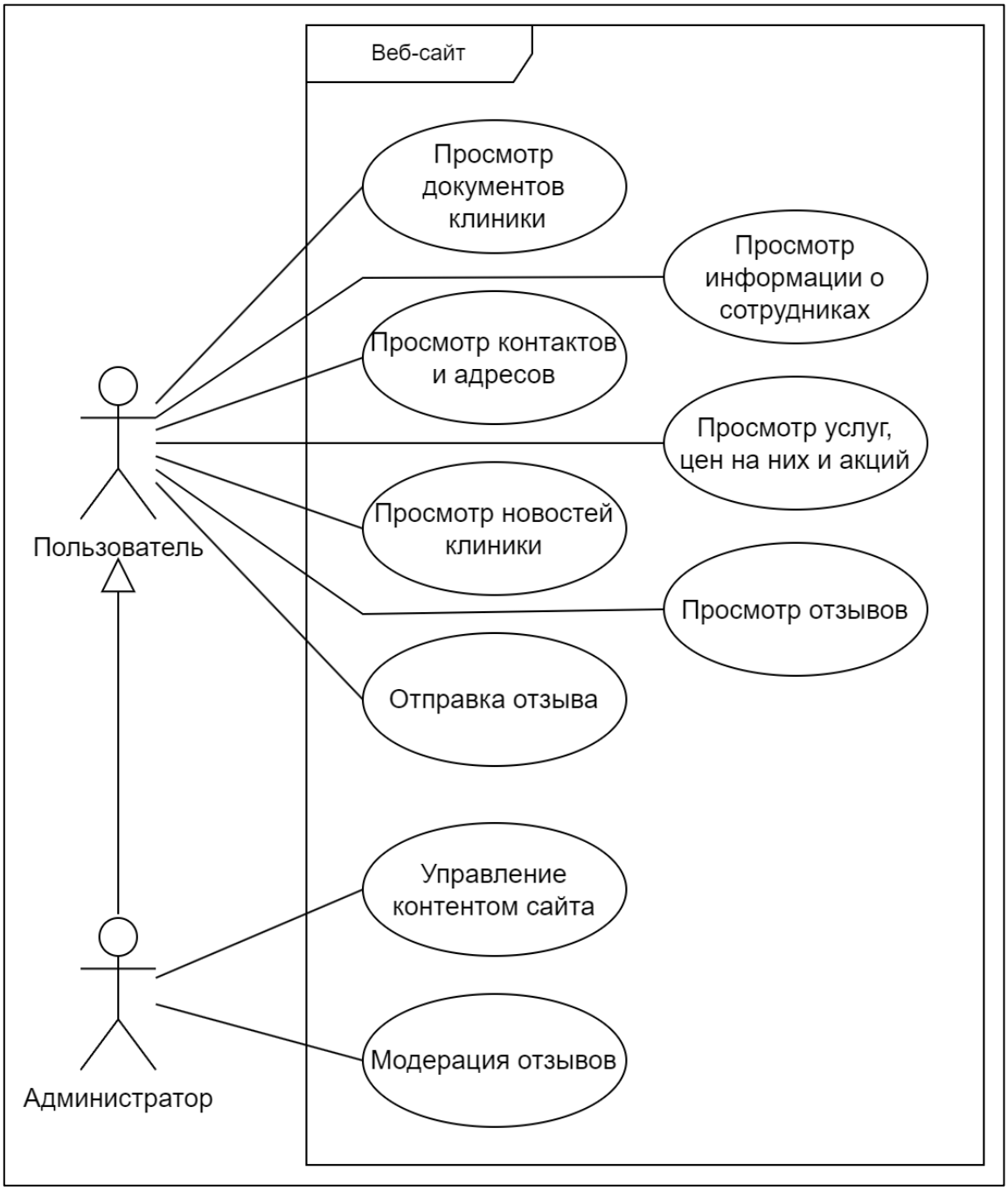

Рисунок 5 – Диаграмма вариантов использования системы

Актер «Пользователь» имеет следующие варианты использования системы.

1. Вариант использования «Просмотр документов клиники» позволяет пользователю просмотреть все открытые для пользователя документы. Документы могут включать в себя политики, справки или соглашения.

2. Вариант использования «Просмотр информации о сотрудниках» позволяет пользователю просмотреть информацию обо всех сотрудниках клиники. Информация о сотрудниках должна включать в себя сведения об образовании, должностях, опыте. Также должна быть предоставлена фотография сотрудника.

3. Вариант использования «Просмотр контактов и адресов» позволяет пользователю просмотреть адреса и контакты клиник. Контакты и адреса должны быть размещены на отдельной странице.

4. Вариант использования «Просмотр новостей клиники» позволяет пользователю просмотреть на главной странице новости и события, опубликованные на веб-сайте.

5. Вариант использования «Просмотр отзывов» позволяет пользователю просмотреть отзывы на услуги и врачей клиники.

6. Вариант использования «Отправка отзывов» позволяет пользователю отправлять отзывы на услуги и врачей клиники.

Актер «Администратор» наследует варианты использования актера «Пользователь» и имеет следующие варианты использования системы.

1. Вариант использования «Управление контентом сайта» позволяет администратору создавать, просматривать, редактировать и удалять все записи базы данных, определяющих контент сайта: новости, сотрудники, услуги и др.

2. Вариант использования «Модерация отзывов» позволяет администратору проводить модерацию отзывов, оставленных пользователями.

Спецификация ключевых вариантов использования (ВИ) системы приведена в таблицах 1–4 приложения.

2. ПРОЕКТИРОВАНИЕ

2.1. Архитектура веб-сайта

Разрабатываемое приложение следует архитектурному паттерну MVC (Model-View-Controller), используемом для разделения архитектуры приложения на отдельные компоненты для упрощения процесса разработка, модификации и поддержки приложения [6]. Паттерн MVC представлен на рисунке 6.

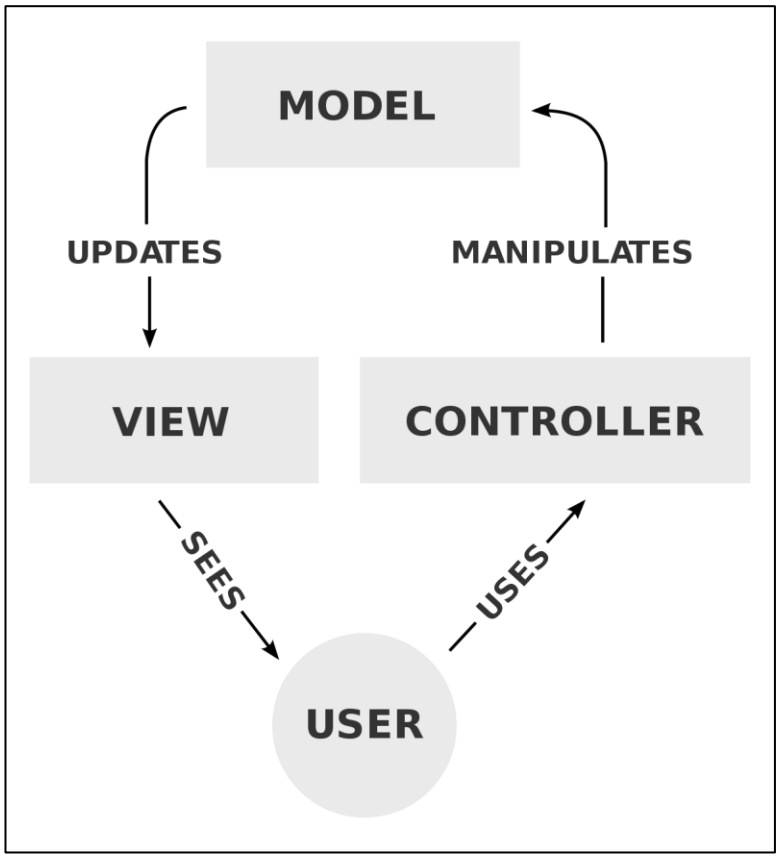

Рисунок 6 – Паттерн Model-View-Controller

Основные компоненты паттерна MVC описаны ниже.

Модель (Model)

Модель представляет собой компонент, отвечающий за доступ к данным приложения. Модель обычно представляет отдельную таблицу в базе данных или какую-либо другую структуру данных. Она определяет правила для взаимодействия с данными, такие как получение, добавление, обновление и удаление.

Вид (View)

Данный компонент отвечает за отображение данных пользователю и обычно представляют собой файлы шаблонов, которые используются для отображения HTML-страниц и динамических данных. Он получает данные от представления и отображают их в удобном для чтения виде для пользователей.

Контроллер (Controller)

Контроллер является посредником между моделью и видом. Он получает данные из модели, обрабатывает их при необходимости и передает в вид для отображения. Основная задача вид – подготовить данные для отображения.

2.2. Проектирование базы данных

Значительную роль в разрабатываемом приложении будет играть база данных и взаимодействие с ней. Поэтому была разработана модель реляционной базы данных [7] для хранения информации о сотрудниках, услугах, клиниках и подобных им объектах.

Модель базы данных была разделена на три части для упрощения просмотра и восприятия.

Первая часть модели базы данных посвящена сотрудникам медицинского центра. Это включает в себя все данные, связанные с персоналом, такие как их личная информация, специализация или график работы.

Вторая часть модели базы данных посвящена услугам и связанным с ними понятиями. Это включает в себя информацию о медицинских услугах, предлагаемых центром, ценах на них и акциях.

Третья часть модели базы данных затрагивает оставшиеся сущности системы, которые не связаны непосредственно с остальной базой данных.

Первая часть схемы базы данных веб-сайта, затрагивающая сотрудников медицинского центра, представлена на рисунке 7.

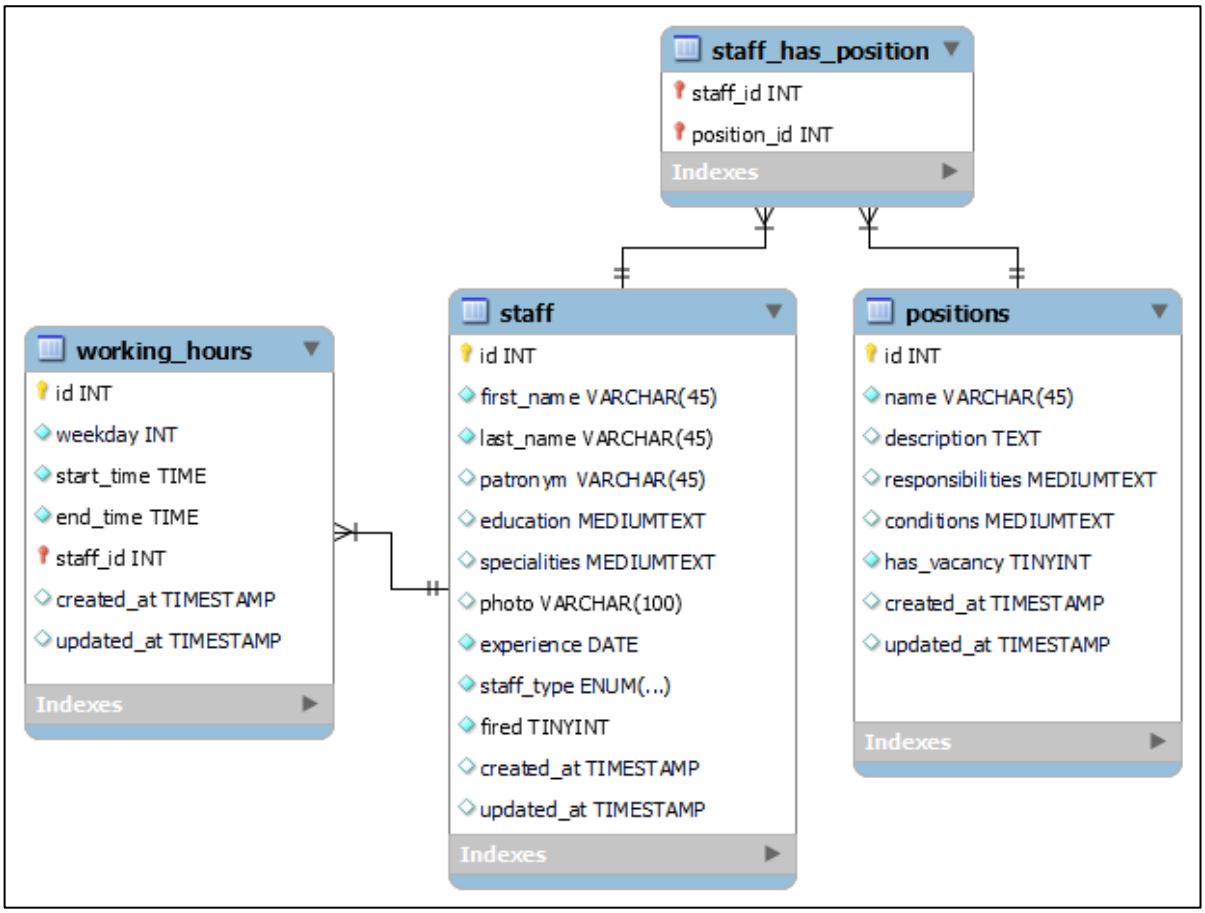

Рисунок 7 – Схема базы данных веб-сайта (часть 1)

Таблица staff содержит информацию о сотрудниках клиники.

Таблица working_hours содержит информацию о рабочих часах сотрудника. Она связана с таблицей staff внешним ключом staff_id.

Таблица positions содержит информацию о должностях сотрудников. Таблицы positions и staff связаны отношением «многие-ко-многим» через таблицу-посредник staff_has_position.

Описание первой части базы данных веб-сайта представлено в таблице 1.

Таблица 1 – Описание схемы базы данных (часть 1)

Название таблицы	**Название поля**	**Тип значения**	**Описание**
staff	first_name	VARCHAR	Имя сотрудника
	last_name	VARCHAR	Фамилия сотрудника
	patronym	VARCHAR	Отчество сотрудника
	education	MEDIUMTEXT	Информация об образовании сотрудника

Окончание таблицы 1

Название таблицы	Название поля	Тип значения	Описание
staff	specialities	MEDIUMTEXT	Информация о специализациях сотрудника
	photo	VARCHAR	Путь к фотографии сотрудника
	experience	DATE	Дата, от которой отсчитывается стаж работы сотрудника
	staff_type	ENUM	Тип сотрудника: врач, медсестра/медбрат, руководство
	fired	TINYINT	Флаг увольнения сотрудника
positions	name	VARCHAR	Название должности
	description	TEXT	Описание должности
	responsibilities	MEDIUMTEXT	Обязанности сотрудника
	conditions	MEDIUMTEXT	Условия работы на должности
	has_vacancy	TINYINT	Флаг существования вакансии для должности
working_hours	weekday	INT	Номер дня недели от 0 до 6
	start_time	TIME	Время начала работы
	end_time	TIME	Время конца работы
	staff_id	INT	Внешний ключ сотрудника

Вторая часть схемы базы данных веб-сайта, затрагивающая услуги медицинского центра, представлена на рисунке 8.

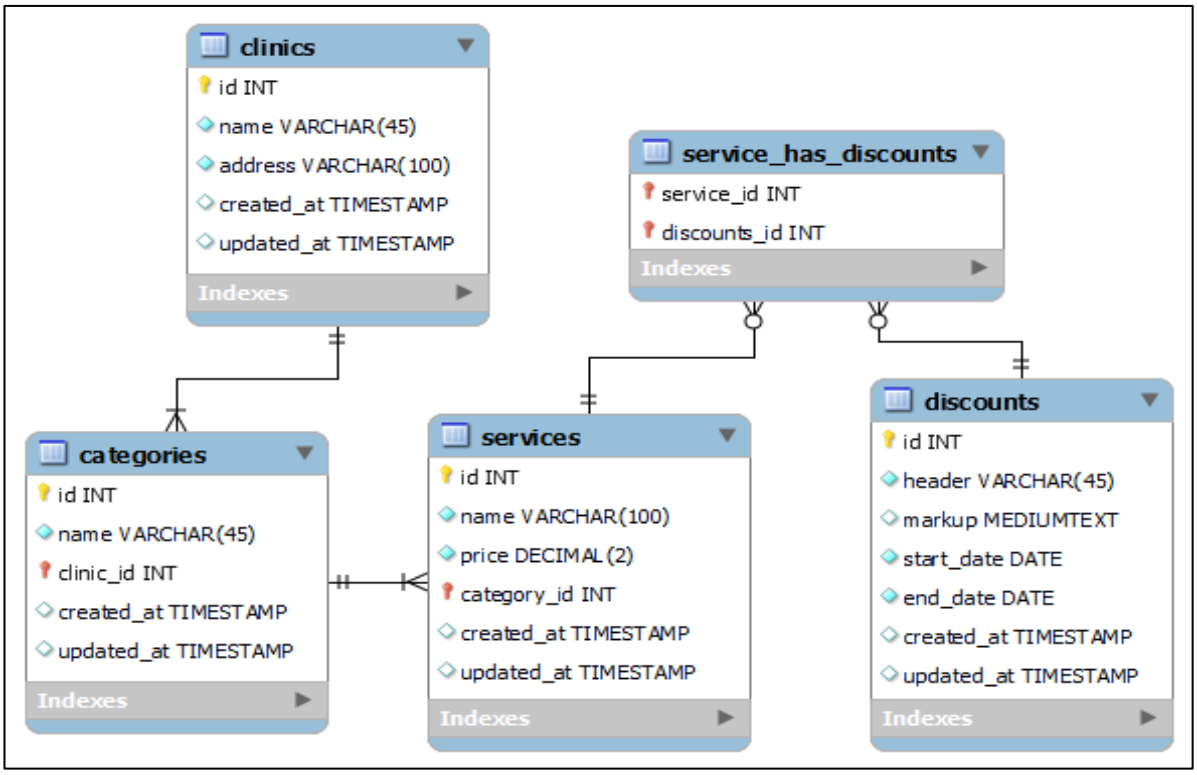

Рисунок 8 – Схема базы данных веб-сайта (часть 2)

Таблица clinics содержит информацию о добавленных в систему клиниках.

Таблица categories содержит информацию о категориях (или направлениях) услуг, оказываемых в клинике. Категории привязываются к клиникам, поэтому эта таблица связана с таблицей clinics внешним ключом clinic_id.

Таблица services содержит всю информацию об оказываемых в медицинском центре услугах. Они привязываются к категориям услуг, поэтому таблица services связана с таблицей categories внешним ключом category_id.

Таблица discounts содержит информацию о скидках и акциях на оказываемые в медицинском центре услуги. Таблицы discounts и services связаны отношением «многие-ко-многим» через таблицу-посредник service_has_discount.

Описание второй части базы данных приведено в таблице 2.

Таблица 2 – Описание схемы базы данных (часть 2)

Название таблицы	**Название поля**	**Тип значения**	**Описание**
clinics	name	VARCHAR	Название клиники
	address	VARCHAR	Адрес клиники
categories	name	VARCHAR	Название категории
	clinic_id	INT	Внешний ключ клиники
services	name	VARCHAR	Название услуги
	price	DECIMAL	Цена услуги
	category_id	INT	Внешний ключ категории
discounts	header	VARCHAR	Заголовок акции
	markup	MEDIUMTEXT	Текстовое содержание акции
	start_date	DATE	Дата начала акции
	end_date	DATE	Дата конца акции

Третья часть схемы базы данных веб-сайта, затрагивающая остальные сущности медицинского центра, представлена на рисунке 9.

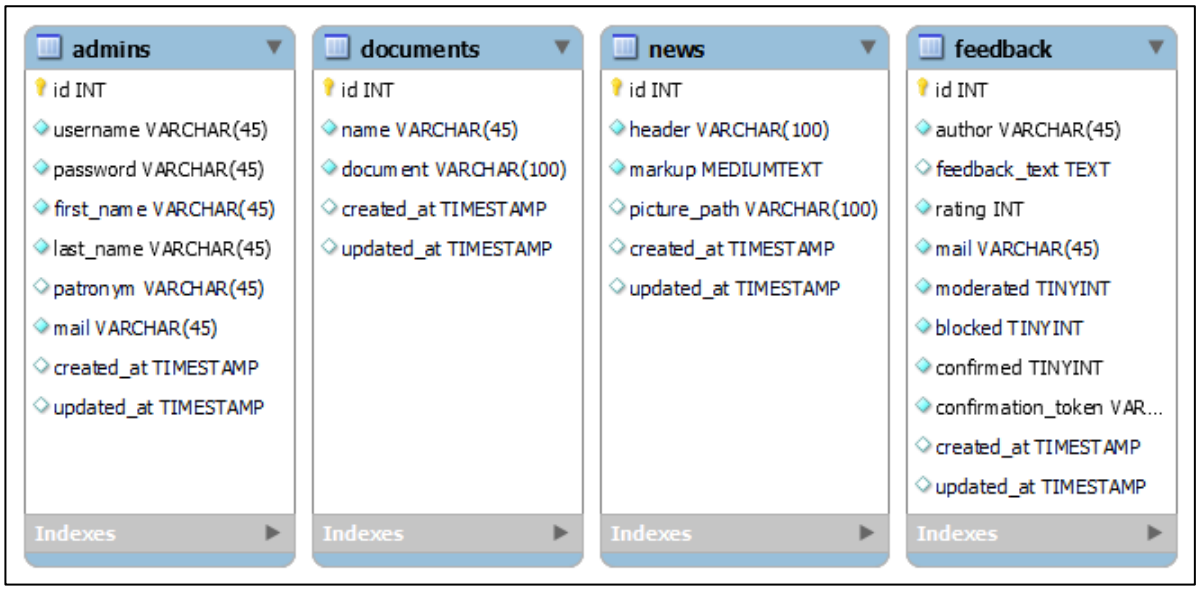

Рисунок 9 – Схема базы данных веб-сайта (часть 3)

Таблица admins содержит информацию об администраторах сайта.

Таблица feedback хранит отзывы пользователей на обслуживание в клинике.

Таблица documents содержит информацию о документах клиники: политиках, приказах, регламентах.

Таблица news хранит новости, отображаемые на соответствующей странице сайта.

Описание третьей части базы данных приведено в таблице 3.

Таблица 3 – Описание схемы базы данных (часть 3)

Название таблицы	**Название поля**	**Тип значения**	**Описание**
admins	username	VARCHAR	Имя администратора, используемое при аутентификации
	password	VARCHAR	Пароль администратора
	first_name	VARCHAR	Имя администратора
	last_name	VARCHAR	Фамилия администратора
	patronym	VARCHAR	Отчество администратора
	mail	VARCHAR	Адрес электронной почты администратора
documents	name	VARCHAR	Название документа
	document	VARCHAR	Путь к файлу документа
news	header	VARCHAR	Заголовок новости
	markup	MEDIUMTEXT	Текстовое содержание новости

Окончание таблицы 3

Название	Название поля	Тип значения	Описание
news	picture_path	VARCHAR	Путь к прикрепленной картинке
feedback	author	VARCHAR	Автор отзыва
	feedback_text	TEXT	Текст отызва
	rating	INT	Дата начала акции
	mail	VARCHAR	Дата конца акции
	moderated	TINYINT	Флаг прошедшей модерации
	blocked	TINYINT	Флаг факта о блокировке администратором
	confirmed	TINYINT	Флаг факта, что отзыв подтвержден пользователем.
	confirmation_token	VARCHAR	Токен, используемый для подтверждения отзыва.

Каждая таблица, кроме тех, которые связывают отношением «многие к многим», содержит поля created_at, updated_at и id. Первые два необходимы для обозначения времени и даты создания и последнего обновления записи в базе данных, третий – для присвоения уникального идентификатора записи в базе данных.

2.3. Проектирование интерфейса веб-сайта

В ходе проектирования интерфейса веб-сайта были разработаны макеты основных страниц проектируемого веб-сайта. Макет – предварительный образец веб-страницы до ее верстки, на котором отмечаются положения основных элементов страницы. Он служит визуальным руководством для расположения и взаимодействия элементов дизайна на странице.

Макеты страниц разрабатываемого веб-сайта были созданы при помощи бесплатного онлайн редактора Figma [8]. Он предлагает широкий спектр функций, включая векторное редактирование, прототипирование и совместное использование документов.

Макет главной страницы проектируемого веб-сайта представлен на рисунке 10.

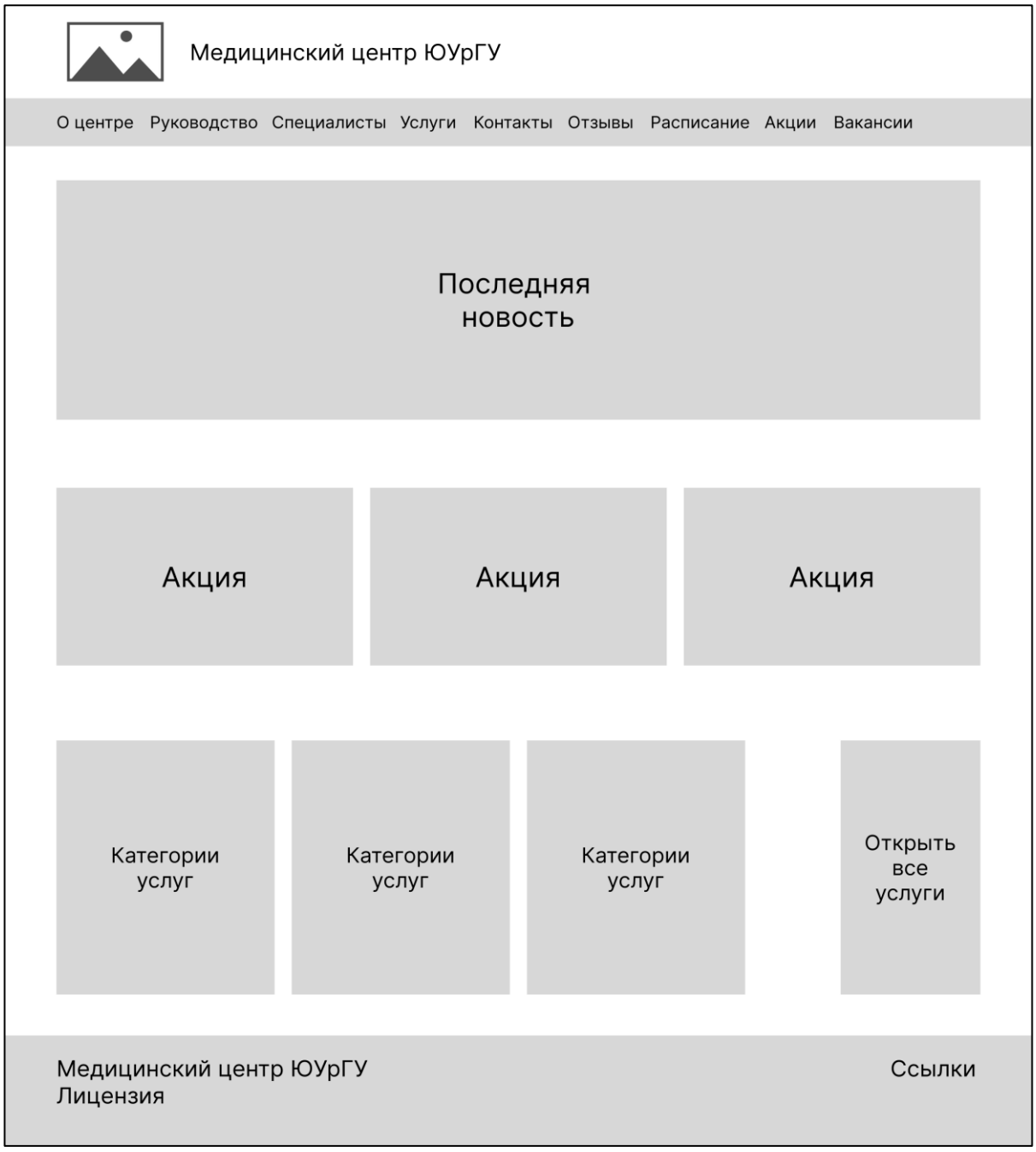

Рисунок 10 – Макет главной страницы веб-сайта

Главная страница веб-сайта содержит ссылки на страницы, располагающие информацией, которая может заинтересовать пользователя: общая информация о центре, руководство и специалисты центра, услуги, контакты центра, отзывы других пользователей, расписание врачей, акции и открытые вакансии. Также пользователю открыты последние новости центра в виде горизонтальной полосы, которую можно проматывать, свежие акции, и список категорий услуг с ссылкой на полный список.

Одним из требований к проектируемому веб-сайту является возможность просмотра информации о сотрудниках медицинского центра. Макет страницы со списком сотрудников представлен на рисунке 11.

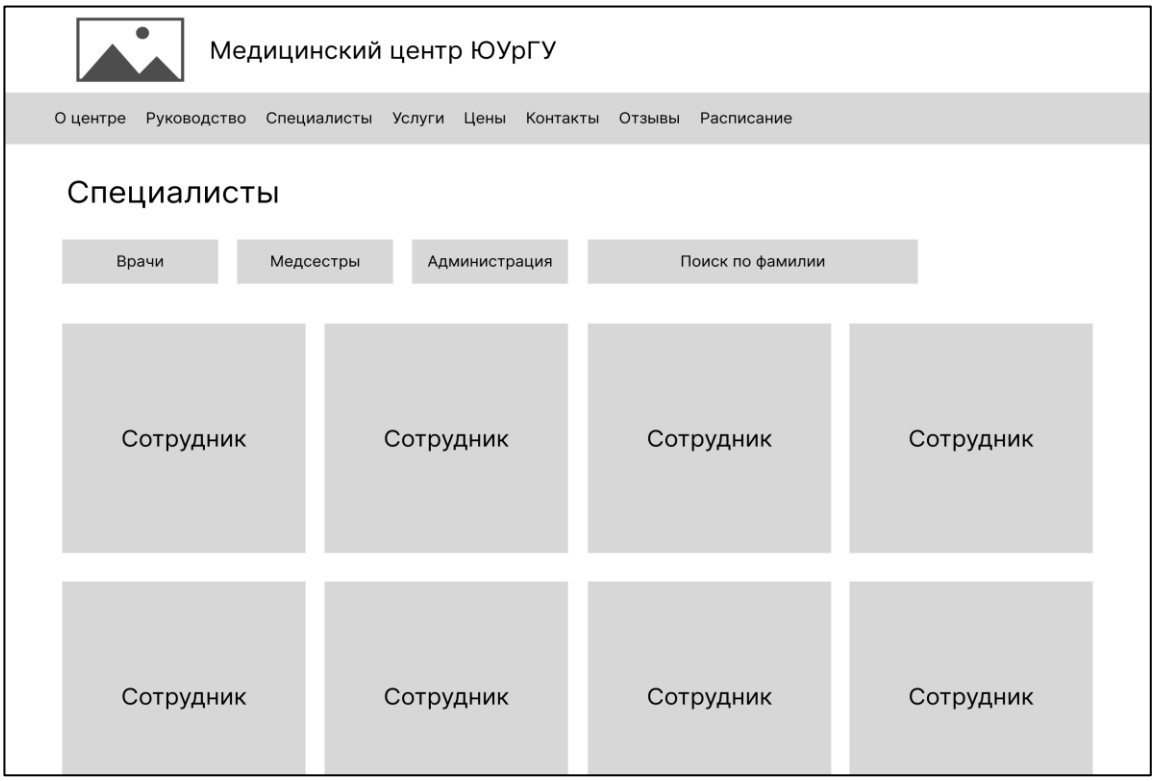

Рисунок 11 – Макет страницы сотрудников

На данной странице перечислены все сотрудники центра, занесенные в базу данных. Краткая информация о каждом сотруднике будет представлена в карточках. К ней относятся: фотография сотрудника, фамилия, имя и отчество сотрудника; занимаемые ими должности. Также в карточке находится ссылка на страницу с полной информацией о сотруднике. Под навигационной панелью находится фильтр списка: категории сотрудников и поисковая строка.

2.4. Алгоритм отправки отзыва

Согласно требованиям к проектируемой системе, она должна предоставлять функцию отправки отзыва пользователем. Для описания процесса отправки отзыва была разработана диаграмма деятельности, которая представлена на рисунке 12.

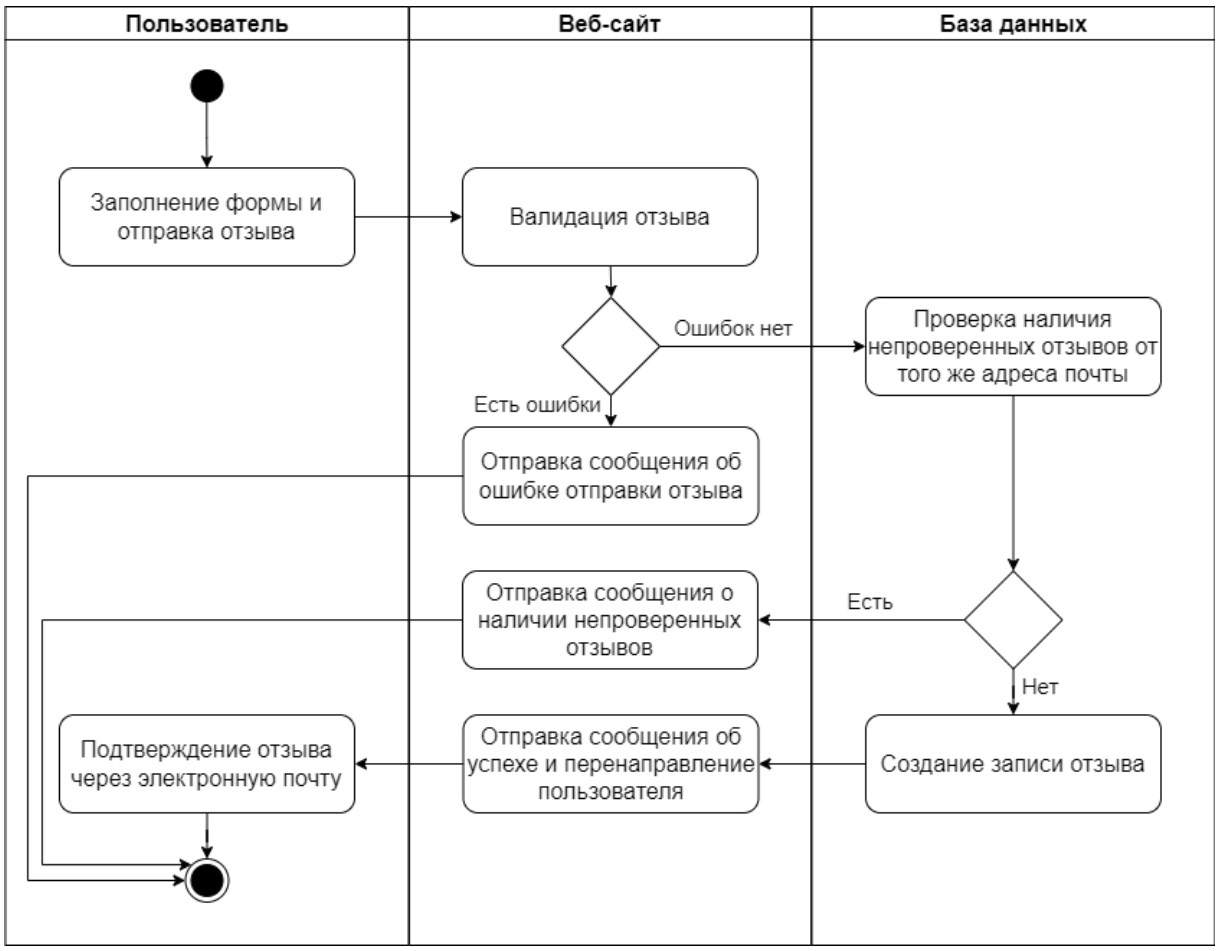

Рисунок 12 – Диаграмма деятельности «Отправка отзыва»

Отправка отзыва начинается с заполнения формы отзыва и ее сохранения. После этого веб-приложение проводит валидацию данных, отправленных в форме. Если она проходит неудачно, веб-приложение отправляет ошибки, которые были выявлены при ее выполнении. Если ошибок нет, создается запрос в базу данных, чтобы узнать, есть ли непроверенные отзывы с таким же адресом почты. Если такие есть, веб-приложение тоже отправляет ошибку пользователю. В противном случае в базе данных создается новая запись отзыва, а веб-приложение отправляет сообщение об успешной отправке отзыва. В конце пользователь получает сообщение об успехе или ошибке, после чего он может подтвердить отзыв через указанный почтовый ящик, если отправка была успешна.

3. РЕАЛИЗАЦИЯ

3.1. Выбор инструментов и технологии

Для реализации разрабатываемой системы были выбраны следующие средства разработки: веб-фреймворк Laravel, язык программирования JavaScript, язык разметки HTML, каскадные таблицы стилей CSS, система управления базами данных MySQL.

Laravel – это веб-фреймворк, написанный на языке программирования PHP [9]. Laravel использует архитектуру MVC, что облегчает организацию кода и делает его более удобным для восприятия. Также он использует Eloquent ORM (Object Relational Mapping) для использования синтаксиса PHP для запросов в базу данных и облегчения работы с ней вместе с языком шаблонов Blade для генерации ответов на запросы от клиента.

JavaScript представляет собой язык программирования, используемый для создания интерактивных веб-страниц [10].

HTML (Hypertext Markup Language) – стандартизированный язык разметки, используемый для создания веб-страниц [11].

CSS (Cascading Style Sheets) – язык иерархических правил или таблиц стилей, позволяющий задать визуальное отображение элементов, определенных в коде разметки [11].

MySQL – реляционная СУБД, одна из самых популярных на данный момент [12]. MySQL имеет ряд преимуществ, включая высокую скорость работы, надежную систему безопасности, наличие бесплатной лицензии и простоту использования.

Bootstrap – бесплатный набор инструментов для разработки пользовательского интерфейса веб-сайтов [13]. Bootstrap облегчает создание адаптивного интерфейса веб-сайта и предлагает готовые CSS-шаблоны для оформления. В реализации разрабатываемого проекта будет использоваться последняя версия Bootstrap 5.

Библиотека jQuery на языке JavaScript, облегчающая взаимодействие с содержимым HTML-документов [14].

3.2. Реализация базы данных

Реализация архитектуры MVC

В Laravel паттерн MVC реализуется следующим образом. Модели в Laravel представляют собой классы, которые взаимодействуют с базой данных. Эти классы расположены в файлах в директории проекта «app\Models», по одному файлу на модель. Они содержат методы для извлечения, вставки и обновления информации в базе данных. Каждая модель соответствует таблице в базе данных.

В разрабатываемом веб-сайте виды представлены в виде файлов определенного расширения. Каждый из них содержит смесь HTML-разметки и специального синтаксиса, помогающего управлять генерацией разметки страницы. Виды расположены в директории «resources\views» проекта.

Контроллеры в разрабатываемом приложении являются классами, содержащими функции обработки запросов. Когда сервер принимает запрос от клиента, запускается соответствующая HTML-запросу функция. Каждый класс расположен в отдельном файле в директории «app\Http\Controllers» проекта.

В контексте Laravel контроллер обрабатывает запрос, использует модель для обработки данных, а затем отображает представление, передавая ему данные.

Модели

Laravel включает в себя Eloquent [9] ORM (object-relational mapper), упрощающий действия с базой данных. ORM – технология связывающая базу данных с концепциями объектно-ориентированного программирования. При использовании Eloquent, каждая таблица базы данных имеет свою модель, которая впоследствии используется для взаимодействия с этой таблицей. Модель используется как для получения записей из базы данных, так для их создания, редактирования, удаления. В архитектуре MVC модели Laravel представляют сущность Model.

Для примера, сотрудники в базе данных представлены моделью Staff, которая представлена в листинге 1.

Листинг 1 – Модель Staff

```
class Staff extends Model
{
    use HasFactory;

    protected $fillable = [
        'first_name',
        'last_name',
        'patronym',
        'specialities',
        'photo_path',
        'experience',
    ];
    public function workingHours(): HasOne
    {
        return $this->hasOne(WorkingHours::class);
    }
    public function positions(): BelongsToMany
    {
        return $this->belongsToMany(Position::class,
'staff_positions_junc', 'staff_id', 'position_id');
    }
    protected $table = 'staff';
}
```

Класс Staff наследуется от класса Model, содержащего основной функционал модели. Атрибут $table указывает на название таблицы в базе данных, которой соответствует данная модель. Также в модели определен атрибут $fillable, указывающий, какие поля таблицы могут быть использованы для заполнения. Также в модели присутствуют функции workingHours и positions, определяющие соответствующие связи с другими таблицами в базе данных.

Миграции

Для управления базой данных удобно использовать инструмент миграций в Laravel. Они представляют собой способ определения и изменения схемы базы данных с использованием кода вместо прямого взаимодействия с СУБД или написания SQL-запросов вручную. Пример миграции для создания таблицы staff, представляющей модель Staff показан в листинге 2.

Листинг 2 – Миграция создания таблицы staff

```
return new class extends Migration
{
  /**
    Run the migrations.
  */
  public function up(): void
  {
    Schema::create('staff', function (Blueprint $table) {
      $table->id();
      $table->string('first_name');
      $table->string('last_name');
      $table->string('patronym')->nullable();
      $table->mediumText('specialities');
      $table->string('photo_path')->nullable();
      $table->timestamp('experience');
      $table->enum('staff_type', ['doctor', 'nurse', 'administrator']);
      $table->timestamps();
      });
  }
  /**
    Reverse the migrations.
  */
  public function down(): void
  {
    Schema::dropIfExists('staff');
  }
};
```

Миграция должна наследоваться от класса `Migration` для корректной работы. Каждая миграция содержит две функции: `up` и `down`. Первая функция используется для запуска изменений, предписанных миграцией, в то время как функция `down` – для их отката. В данном примере функция `up` создает таблицу staff вместе со всеми необходимыми для нее полями, а `down`, соответственно, удаляет таблицу.

3.3. Организация навигации на веб-сайте

Взаимодействие клиента и сервера происходит посредством HTTP-запросов. Запрос, посылаемый клиентом, имеет метод, который определяет какое желаемое действие выполнится для данного ресурса. В разрабатываемом проекте используются следующие методы: GET, POST, PUT, PATCH и DELETE.

Маршруты в Laravel – способ определения того, как HTTP-запросы обрабатываются веб-приложением. Каждый маршрут ассоциируется с

определенным URL-адресом и методом (или методами) HTTP. Когда пользователь делает запрос сайту, Laravel сопоставляет вводимый пользователем URL-адрес и метод HTTP этого запроса, а затем ищет соответствующий маршрут в приложении. Если такой маршрут был найден, Laravel выполняет код, связанный с этим маршрутом, который обычно включает вызов определенного метода в контроллере.

Все маршруты, которые может использовать клиент, перечислены в файле «web.php» в папке «routes» корневой директории проекта. Каждый маршрут принимает строку с запросом и имя контроллера, который содержит функцию, обрабатывающую указанный запрос [9]. Примеры маршрутов разрабатываемого приложения представлены в листинге 3.

Листинг 3 – Примеры маршрутов

```
Route::resources([
    'admin/resources/events' => EventController::class,
    'admin/resources/documents' => DocumentController::class,
    'admin/resources/clinics' => ClinicController::class,
    'admin/resources/categories' => CategoryController::class,
    'admin/resources/services' => ServiceController::class,
    'admin/resources/discounts' => DiscountsController::class,
    'admin/resources/staff' => StaffController::class,
    'admin/resources/working-hours' => ClinicController::class,
    'admin/resources/positions' => PositionController::class,
    'admin/resources/vacancies' => VacancyController::class,
    'admin/resources/users' => UserController::class,
]);
Route::get('/home', [HomeController::class, 'index'])->name('home');
Route::get('/admin', [AdminController::class, 'dashboard'])
        ->name('dashboard');
Route::get('/admin/resources', [AdminController::class, 'resources'])
        ->name('resources');
Route::get('/admin/moderation')->name('moderation');
Route::get('/admin/login', [LoginController::class, 'showLoginForm'])
        ->name('login');
Route::post('/admin/login', [LoginController::class, 'login'])
        ->name('login');
Route::get('/admin/logout', [LoginController::class, 'logout'])
        ->name('logout');
```

Для определения маршрутов используется класс `Route`, представляющий статический интерфейс для взаимодействия с функционалом Laravel. Метод `resources` позволяет легко создать маршруты для взаимодействия с ресурсами проекта, которые определяют контент веб-сайта. Он автоматически создает все запросы, необходимые для создания, чтения, редактиро-

вания и удаления записей в базе данных. Методы `get` и `post` позволяют, соответственно, создать маршруты методов GET и POST.

3.4. Обработка запросов клиента

Контроллеры в Laravel представляют собой классы, которые содержат логику обработки HTTP-запросов. Они служат промежуточным звеном между видами и моделями, обрабатывая запросы от пользователей и возвращая ответы. В архитектуре MVC контроллеры Laravel представляют сущность Controller.

Каждый контроллер имеет ряд методов, которые содержат код обработки запросов к приложению. Для работы написанного метода он должен быть указан в созданном маршруте в файле «web.php». Примеры методов контроллера `StaffController`, предназначенного для обработки запросов CRUD к таблице staff, представлены в листинге 4.

Листинг 4 – Примеры методов обработки запросов

```
public function index()
{
    return view('resources.staff.index', [
        'staff' => Staff::query()->orderBy('staff_type')
                ->orderBy('last_name')->orderBy('first_name')
                ->orderBy('patronym'),
    ]);
}
public function store(Request $request)
{
    $validated_data = $request->validate([
        'first_name' => ['required', 'max:255'],
        'last_name' => ['required', 'max:255'],
        'patronym' => ['max:255', 'string'],
        'specialities' => ['required', 'max:16777215'],
        'experience' => ['required', 'date', 'before:now'],
        'photo_path' => ['file', 'image'],
        'staff_type' => ['required', 'in:doctor,nurse,administrator'],
        'positions' => ['array'],
        'positions.*' => ['exists:positions,id'],
    ]);
    $staff = Staff::create([
        'first_name' => $validated_data['first_name'],
        'last_name' => $validated_data['last_name'],
        'patronym' => $validated_data['patronym'],
        'specialities' => $validated_data['specialities'],
        'experience' => $validated_data['experience'],
        'staff_type' => $validated_data['staff_type']
    ]);
    $path = $request->file('photo_path')->storeAs('staff_photos',
"staff{$staff->id}" . '.' . $request->file('photo_path')->getExtension());
    $staff->photo_path = $path;
```

```
    $staff->positions()->attach($validated_data['positions']);
    $staff->save();
    return redirect('/admin/resources/staff');
}
public function destroy(string $id)
{
    $staff = Staff::query()->findOrFail($id);
    Storage::delete($staff->photo_path);
    $staff->delete();
    return redirect('/admin/resources/staff');
}
```

В данном примере представлены 3 метода. Метод `index` предназначен для вывода списка всех сотрудников в базе данных. Он возвращает вид по указанному пути и передает туда массив из данных для отображения, в которой содержится только коллекция услуг, полученных из базы данных при помощи модели `Staff`.

Метод `store` предназначен для сохранения записи сотрудника в базе данных. Вместе с POST запросом на сервер подаются данные создаваемого, которые проходят валидацию при помощи метода `validate`. Если валидация данных прошла успешно, создается новая запись сотрудника медицинского центра, к которой привязываются указанные должности и загруженная фотография, а пользователь перенаправляется на страницу со списком сотрудников. Если данные не проходят валидацию, пользователь перенаправляется обратно на форму создания, где ему открываются все выявленные ошибки.

Метод `destroy` предназначен для удаления записи сотрудника. Вместе с запросом передается идентификатор удаляемого сотрудника, по которому при помощи модели находится искомая запись и удаляется.

3.5. Реализация пользовательского интерфейса

Для разработки пользовательского интерфейса Laravel предлагает язык шаблонов Blade, который позволяет генерировать содержание страницы сайта при помощи смеси специального синтаксиса и кода HTML. Помимо всего прочего, Blade позволяет использовать чистый PHP код для управления логикой генерации ответа.

Также в реализации интерфейса используется набор инструментов Bootstrap5, который обеспечивает адаптивность интерфейса в разрабатываемом веб-сайте при помощи системы сеток.

Пример реализации пользовательского интерфейса представлен на рисунке 13. На нем показана главная страница разрабатываемого веб-сайта.

Рисунок 13 – Главная страница веб-сайта

Еще один пример реализации пользовательского интерфейса представлен на рисунке 14. На нем показана страница отзывов.

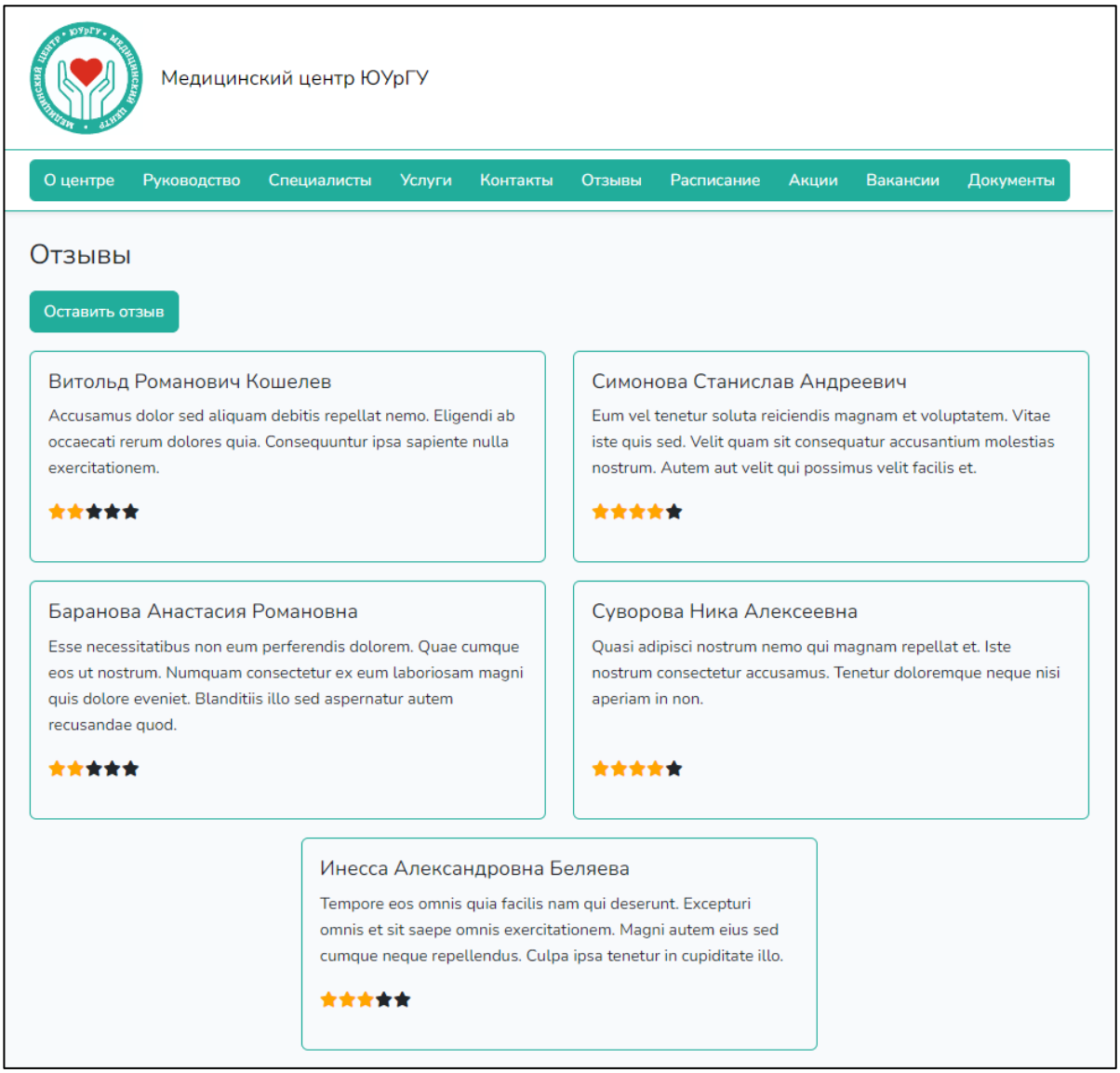

Рисунок 14 – Страница отзывов

Язык шаблонов Blade содержит набор директив для управления логикой генерации страницы. Директивы Blade – сокращения обычных конструкций языка PHP, созданные для удобства разработчика. Директивы начинаются с символа «@», за которым следует название директивы и любые другие необходимые параметры. Пример использования директив Blade можно посмотреть на примере фрагмента с акциями с главной страницы веб-сайта (листинг 5).

Листинг 5 – Фрагмент с акциями на главной странице

```
<div class="row justify-content-center mb-5">
  @foreach ($discounts as $discount)
    <div class="col-md d-flex mb-3 mb-md-0">
      <div class="card flex-fill">
```

```
        <h5 class="card-header">Акция</h5>
          <div class="card-body d-flex flex-column">
            <h5 class="card-title">{{ $discount->header }}</h5>
            <p class="card-text flex-grow-1">{{$discount->content}}</p>
              @if ($discount->end_date)
                @php
                  $end_date_string = Carbon::createFromFormat('Y-m-
d',$discount->end_date)->format('d.m.Y');
                @endphp
                <p class="card-text"><small class="text-muted">
Акция действует до {{$end_date_string}}</small></p>
              @endif
              <a href="{{ route('discountsShow', $discount->id) }}"
                class="btn btn-primary mt-auto">Просмотреть</a>
          </div>
        </div>
      </div>
  @endforeach
 </div>
```

На представленном листинге можно увидеть директиву `@foreach`, которая создает цикл, который проходит по каждому элементу массива `$discounts`. В нем содержатся объекты акций, запрошенных из базы данных и переданных в представление. Цикл вставляет весь код до `@endforeach` столько раз, сколько элементов находится в массиве. При помощи директивы `@if` создается условие, при выполнении которого будет выполнен код от `@if` до `@endif`. Директива `@php` позволяет использовать чистый PHP код, например, для инициализации переменной с отформатированной датой, которая будет затем использоваться в выводе, как показано на рисунке.

Для отображения значений переменных в Blade используется конструкция из двойных фигурных скобок. Таким образом, выражение `{{$discount->header}}` из примера выведет заголовок акции, а `{{route('discountsShow', $discount->id)}}` выведет ссылку на страницу с более подробной информацией об акции.

Для демонстрации адаптивности пользовательского интерфейса, раздел с акциями на главной странице при малом размере экрана можно представлен на рисунке 15.

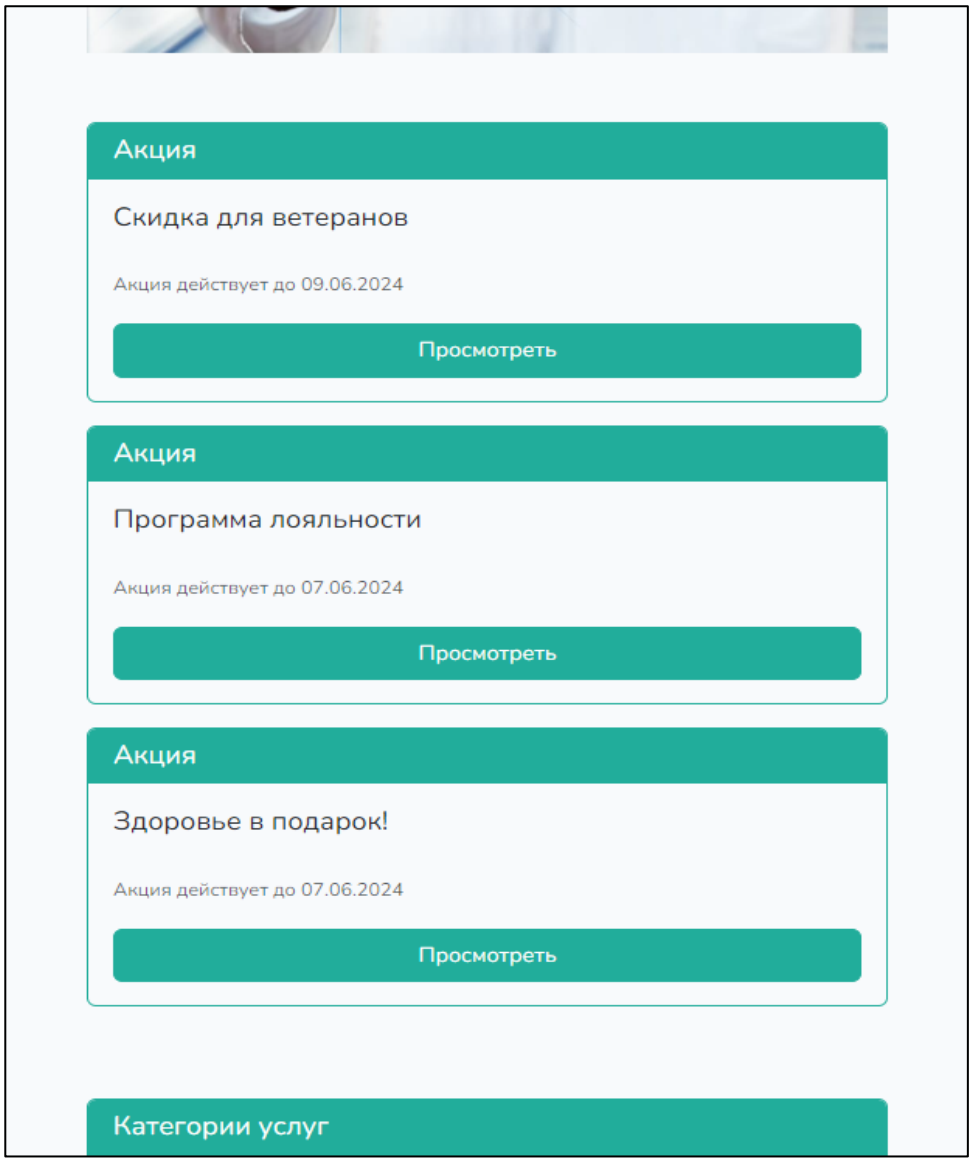

Рисунок 15 – Отображение акций на главной странице

Адаптивность интерфейса обеспечивается системой сеток Bootstrap. Сетка – композиция из рядов и двенадцати колонок, в которые вставляются элементы интерфейса. При достаточно узком дисплее, колонки переставляются вертикально, чтобы обеспечить удобство при просмотре. В листинге 5, весь раздел с акциями расположен в блоке с CSS классом `row`, который определяет ряд. Ряд содержит блоки с классом `col-md`, который определяет колонку. В названии класса «md» обозначает точку, при которой колонки переставляются вертикально – ширина в 720 пикселей. Когда ширина дисплея уменьшается до 720 пикселей, карты с акциями переместятся так, чтобы они располагались вертикально.

4. ТЕСТИРОВАНИЕ

Одним из важных этапов разработки программного продукта является его тестирование. Для разработанной системы было проведено функциональное тестирование.

Функциональное тестирование – метод тестирования программного продукта, который проверяет соответствие продукта функциональным требованиям. Оно фокусируется на тестировании функциональности системы и ее компонентов с целью убедиться, что они работают правильно и взаимодействуют между собой так, как ожидается [15].

Набор тестов реализованного веб-сайта представлен в таблице 4.

Таблица 4 – Результаты функционального тестирования

№	Название теста	Шаги	Ожидаемый результат	Тест пройден?
1	Просмотр общей информации о медицинском центра.	Нажать на кнопку навигационной панели: «О центре».	На странице выводится информация, предварительно составленная администратором.	Да
2	Просмотр контактов медицинского центра.	Нажать на кнопку навигационной панели: «Контакты».	На странице выводится информация, предварительно составленная администратором.	Да
3	Просмотр лицензии медицинского центра.	Нажать на ссылку в низу страницы «Лицензия».	При просмотре лицензии в браузере пользователя открывается соответствующий файл.	Да
4	Просмотр информации о сотрудниках медицинского центра.	1. Нажать на кнопку «Специалисты» навигационной панели. 2. Выбрать необходимую категорию специалиста или ввести фильтр списка и нажать на «Обновить».	После открытия страницы раскрывается список всех сотрудников центра с пагинацией. При использовании поиска отсеиваются ненужные сотрудники.	Да
5	Просмотр предоставляемых медицинским центром услуг.	1. Нажать на кнопку «Услуги» навигационной панели. 2. Выбрать отделение центра, услуги которого необходимы. 3. Выбрать необходимую категорию услуг.	Открывается список всех услуг выбранной категории вместе с ценами на них.	Да

Продолжение таблицы 4

№	Название теста	Шаги	Ожидаемый результат	Тест пройден?
6	Просмотр акций на услуги медицинского центра	Нажать на кнопку «Акции» навигационной панели.	На странице открывается список всех действующих на данный момент акций со сроками действия.	Да
7	Просмотр вакансий медицинского центра	1. Нажать на кнопку «Вакансии» навигационной панели. 2. В открывшемся списке выбрать должность и нажать «Подробнее».	При переходе на странице вакансий открывается список всех открытых вакансий. При нажатии на «Подробнее» открывается новая страница с более подробной информацией о вакансии.	Да
8	Просмотр отзывов о клинике.	Нажать на кнопку «Отзывы» на навигационной панели.	Открывается список отзывов, отсортированных от новых к старым. Присутствует пагинация.	Да
9	Отправка отзыва.	1. Нажать на кнопку «Отзывы» на навигационной панели, затем нажать на «Оставить отзыв». 2. Заполнить форму отзыва. 3. Нажать на кнопку «Сохранить». 4. Подтвердить отзыв в письме, отправленном на адрес электронной почты пользователя.	После подтверждения в базе данных появляется запись отправленного отзыва, а на сохраненный в системе адрес электронной почты приходит уведомление о новом отзыве. В окне модерации на панели администратора появляется отзыв, ожидающий проверки.	Да
10	Отправка отзыва при неподтвержденном или непроверенном отзыве	Выполнить все шаги отправки отзыва, кроме подтверждения.	После попытки сохранить отзыв пользователю открывается сообщение об ошибке.	Да
11	Просмотр документов медицинского центра.	1. Нажать на кнопку «Документы». 2. Выбрать документ из списка и нажать на «Просмотреть».	В браузере пользователя открывается файл выбранного документа.	Да
12	Аутентификация администратора.	1. Перейти на панель администратора. 2. Ввести подлинные имя пользователя и пароль и нажать на «Войти».	После входа открывается панель администратора.	Да

Продолжение таблицы 4

№	Название теста	Шаги	Ожидаемый результат	Тест пройден?
13	Создание ресурса.	1. Перейти на панель администратора. 2. Перейти на страницу ресурсов. 3. Выбрать в списке требуемый ресурс. 4. Нажать на кнопку «Просмотреть». 5. Нажать на кнопку «Создать». 6. Заполнить форму и нажать на «Сохранить».	В базе данных и в списке ресурса появляется только что созданная запись.	Да
14	Редактирование ресурса.	1. Перейти на панель администратора. 2. Перейти на страницу ресурсов. 3. Выбрать в списке требуемый ресурс. 4. Нажать на кнопку «Просмотреть». 5. Нажать на кнопку «Редактировать». 6. Заполнить форму и нажать на «Сохранить».	В базе данных и в списке ресурса изменяется соответствующая запись.	Да
15	Создание ресурса с некорректными данными.	1. Выполнить все шаги создания ресурса, кроме заполнения и сохранения формы. 2. Ввести некорректные данные формы, которые не смогут пройти валидацию. 3. Нажать на кнопку «Сохранить».	Пользователь перенаправляется обратно на страницу с формой ресурса с сообщениями об ошибках валидации.	Да
16	Редактирование ресурса с некорректными данными.	1. Выполнить все шаги редактирования ресурса, кроме заполнения и сохранения формы. 2. Ввести некорректные данные формы, которые не смогут пройти валидацию. 3. Нажать на кнопку «Сохранить».	Пользователь перенаправляется обратно на страницу с формой ресурса с сообщениями об ошибках валидации.	Да

Окончание таблицы 4

№	Название теста	Шаги	Ожидаемый результат	Тест пройден?
17	Модерация отзыва.	1. Перейти на страницу модерации панели администратора. 2. Отфильтровать список отзывов, нажав на «Непроверенные» или «Все». 3. Выбрать непроверенный отзыв из списка и нажать на «Открыть». 4. Нажать на «Заблокировать», «Удалить» или «Опубликовать».	При фильтрации списка отсеиваются соответствующие отзывы. Запись в базе данных рассматриваемого отзыва получает соответствующие изменения, что также сказывается на странице отзывов.	Да
18	Изменение контента веб-сайта	1. Перейти на панель администратора. 2. Перейти на страницу «Контент». 3. Выбрать в списке «О центре» и нажать на «Изменить». 4. Внести корректировки в открывшейся форме и нажать на «Сохранить». 5. Выбрать в списке «Контакты». 6. Повторить шаги 3 и 4.	После сохранения форм, содержания соответствующих страниц («О центре» и «Контакты») изменяется в соответствии со сделанными корректировками.	Да

Таким образом, система была протестирована, все тесты были пройдены успешно.

Также было проведено тестирование запросов разработанного веб-сайта при помощи платформы Postman [16]. Были протестированы 46 запросов веб-сайта, из которых 20 имеют метод GET, 9 – метод POST, 9 – метод PUT и 8 – метод DELETE. Все тесты прошли успешно.

ЗАКЛЮЧЕНИЕ

В рамках данной выпускной квалификационной работы был разработан веб-сайт, представляющий медицинский центр ЮУрГУ. При этом были решены все поставленные задачи, перечисленные ниже.

1. Был проведен анализ аналогичных проектов.
2. Были выполнены проектирование и реализация базы данных веб-сайта.
3. Были выполнены проектирование и реализация веб-сайта.
4. Было проведено тестирование веб-сайта.

Разработанный веб-сайт удовлетворяет всем выявленным функциональным и нефункциональным требованиям. Пользователь имеет возможность получить требуемую информацию в удобном виде, что включает в себя сведения о сотрудниках, услуги и цены на них, акции, последние новости, отзывы других пользователей. Контент веб-сайта редактируется через специальную панель администратора. Через неё же аутентифицированный администратор может проводить модерацию отзывов, блокируя их или публикуя.

Также веб-сайт имеет адаптивный интерфейс, что позволяет взаимодействовать с ним на устройствах с разными размерами дисплея, что обеспечивает удобство использования веб-сайта на любом устройстве, будь то смартфон, планшет или персональный компьютер.

ЛИТЕРАТУРА

1. Клиника «Источник». [Электронный ресурс] URL: https://ci74.ru (дата обращения: 02.02.2024 г.).

2. Медицинский центр «Медеор». [Электронный ресурс] URL: https://medeor74.ru (дата обращения: 02.02.2024 г.).

3. Медицинский центр «Лотос». [Электронный ресурс] URL: https://www.lotos74.ru (дата обращения: 02.06.2024 г.).

4. Приказ Минздрава РФ от 30.12.2014 N 956Н. [Электронный ресурс] URL: https://normativ.kontur.ru/document?moduleId=1&documentId=246823 (дата обращения: 02.06.2024 г.).

5. Буч Г., Рамбо Д., Якобсон И. Введение в UML от создателей языка. 2-е издание – М.: ДМК Пресс, 2010 г. – 496 с.

6. Дронов В.А. Laravel: быстрая разработка динамических Web-сайтов на PHP, MySQL, HTML и CSS. – СПб.: БХВ-Петербург, 2018 – 768 с.

7. Осипов Д. Л. Технологии проектирования баз данных / Д. Л. Осипов. – Москва: ДМК Пресс, 2019. – 498 с.

8. Figma: the collaborative interface design tool. [Электронный ресурс] URL: https://www.figma.com/ (дата обращения: 02.02.2024 г.).

9. Документация Laravel. [Электронный ресурс] URL: https://laravel.com/docs/10.x (дата обращения: 01.02.2024 г.).

10. Современный учебник JavaScript. [Электронный ресурс]. URL: https://learn.javascript.ru/ (дата обращения: 02.02.2024 г.).

11. Дакетт Д. HTML и CSS. Разработка и дизайн веб-сайтов / Джон Дакетт; [пер. с англ. М.А. Райтмана]. – М.: Эксмо, 2013. – 480 с.

12. MySQL. [Электронный ресурс]. URL: https://www.mysql.com/ (дата обращения: 02.02.2024 г.).

13. Bootstrap 5.3. Документация. [Электронный ресурс] URL: https://getbootstrap.com/docs/5.3/getting-started/introduction (дата обращения: 30.05.2024 г.).

14. Документация jQuery API. [Электронный ресурс] URL: https://api.jquery.com (дата обращения: 02.06.2024 г.).

15. Аниче, М. Эффективное тестирование программного обеспечения / М. Аниче; перевод с английского А. Н. Киселева. – Москва: ДМК Пресс, 2023. – 370 с.

16. Postman API Platform. [Электронный ресурс] URL: https://www.postman.com (дата обращения: 11.06.2024 г.).

ПРИЛОЖЕНИЕ. Спецификация вариантов использования

Таблица 1 – Спецификация ВИ «Просмотр информации о сотрудниках»

Прецедент: Просмотр информации о сотрудниках
ID: 1
Краткое описание: Просмотр полной информации о сотрудниках клиники
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: отсутствуют
Основной поток: 1. Прецедент начинается, когда пользователь открывает страницу информации о сотрудниках клиники. 2. Система отправляет ответ в виде страницы со списком сотрудников. 3. Используя фильтры поиска, пользователь находит нужного сотрудника и нажимает на окно с ним. 4. Система отправляет ответ в виде страницы с информацией о сотруднике
Постусловия: 1. Информация о сотруднике выводится на открывшейся странице.
Альтернативные потоки: I. В базе данных отсутствует сотрудник с заданными параметрами. 1. Система выводит сообщение об отсутствии требуемого сотрудника.

Таблица 2 – Спецификация ВИ «Отправка отзыва»

Прецедент: Отправка отзыва
ID: 2
Краткое описание: Публикация отзыва пользователя о своем опыте пребывания в клинике.
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: отсутствуют
Основной поток: 1. Прецедент начинается, когда пользователь открывает страницу отзывов на веб-сайте. 2. Пользователь нажимает на кнопку «Оставить отзыв» и заполняет открытую форму: имя пользователя, адрес электронной почты, рейтинг от 1 до 5 и текст отзыва. 3. Пользователь нажимает на кнопку «Отправить».
Постусловия: 1. В базе данных присутствует запись оставленного отзыва.
Альтернативные потоки: I. Пользователь нажимает на кнопку «Отправить», не заполнив поля имени, адреса электронной почты или рейтинга. 1. Система выводит сообщение о необходимости заполнения выделенных полей.

Таблица 3 – Спецификация ВИ «Модерация отзывов»

Прецедент: Модерация отзывов
ID: 3
Краткое описание: Проверка отзывов администратором с последующим подтверждением отправки или блокировкой отзыва
Главные актеры: Администратор
Второстепенные актеры: Пользователь
Предусловия: отсутствуют
Основной поток: 1. Прецедент начинается, когда пользователь оставляет отзыв на сайте. 2. Администратор заходит на страницу модерации в панели администратора. 3. Администратор открывает последний оставленный отзыв и проверяет его. 4. Администратор нажимает на кнопку «Подтвердить» или «Отклонить».
Постусловия: 1. В базе данных присутствует изменяется соответствующее поле о пройденной модерации.
Альтернативные потоки:

Таблица 4 – Спецификация ВИ «Управление контентом сайта»

Прецедент: Управление контентом сайта
ID: 4
Краткое описание: Создание, чтение, редактирование и удаление записей в базе данных, формирующих контент сайта, через панель администратора.
Главные актеры: Администратор
Второстепенные актеры: Нет
Предусловия: отсутствуют
Основной поток: 1. Прецедент начинается, когда администратор заходит на страницу панели администратора. 2. Администратор заходит на страницу ресурсов сайта и выбирает требуемый ресурс. 3. Администратор выбирает требуемое действие и открывает форму, соответствующую действию. 4. Администратор вводит необходимые данные и нажимает на «Сохранить».
Постусловия: 1. В базе данных присутствует изменяется, создается или удаляется запись, с которой взаимодействовал администратор.
Альтернативные потоки: I. Действие администратора не проходит валидацию данных. 1. Система выводит сообщение ошибок у каждого поля заполняемой формы.