



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ
КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)
НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.01 Информатика и вычислительная техника

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОЙ РАБОТЕ НА ТЕМУ:

Веб-приложение «Сиа» для организации очереди на
обслуживание

Студент ИУ6-52Б
(Группа)

И.И. 02.12.2021
(Подпись, дата)

Р.В. Баканов
(И.О. Фамилия)

Руководитель курсовой работы

Г.С. 02.12.2021
(Подпись, дата)

Г.С. Иванова
(И.О. Фамилия)

*Процесс
проверки выполнен
оценкой 50 баллов*

2021 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ

Заведующий кафедрой ИУ6
(Индекс)

А.В. Пролетарский
(И.О.Фамилия)

« 13 » сентября 2021 г.

ЗАДАНИЕ
на выполнение курсовой работы

по дисциплине Технология разработки программных систем

Студент группы ИУ6-52Б

Баканов Роман Викторович
(Фамилия, имя, отчество)

Тема курсовой работы Веб-приложение для оптими-
зации очереди на обслуживание

Направленность КР (учебная, исследовательская, практическая, производственная, др.)
учебная

Источник тематики (кафедра, предприятие, НИР) кафедра

График выполнения КР: 25% к 4 нед., 50% к 7 нед., 75% к 11 нед., 100% к 14 нед.

Техническое задание см. техническое задание в приложении А

Оформление курсовой работы:

1. Расчетно-пояснительная записка (РПЗ) на 25-30 листах формата А4.
2. Техническое задание на 5-9 листах формата А4 – оформляется в качестве приложения А к РПЗ.
3. Руководство пользователя на 6-8 листах формата А4 – оформляется в качестве приложения Б к РПЗ (если предусмотрено в техническом задании).
4. Графический и иллюстративный материал оформляется в виде рисунков и помещается в РПЗ.
5. Все материалы и исходный текст программы загрузить на страницу дисциплины на сайте кафедры.

Дата выдачи задания « 1 » сентября 2021 г.

Руководитель курсовой работы

13.09.2021
(Подпись, дата)

Г.С. Иванова
(И.О.Фамилия)

Студент

13.09.2021
(Подпись, дата)

Р.В. Баканов
(И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

РЕФЕРАТ

Расчетно-пояснительная записка 61 страница, 3 части, 11 рисунков, 9 таблиц, 5 источников, 2 приложения.

ОНЛАЙН-ОЧЕРЕДЬ, РАСПИСАНИЕ, ОНЛАЙН-ЗАПИСЬ, ВЕБ-ПРИЛОЖЕНИЕ, СОЗДАНИЕ КОМНАТ.

Объектом разработки является веб-приложение «Cua».

Цель работы – готовое веб-приложение, которое можно использовать для создания очередей и расписаний внутри отдельных комнат с последующей возможностью вступления в эти очереди и записи на прием для оптимизации процесса обслуживания клиентов.

В результате работы было спроектировано и реализовано веб-приложение для организации очередей на обслуживание и планирования расписаний приемов, предоставляющее возможность общения в чате внутри комнаты, получения уведомлений о предстоящих событиях и ограничения доступа к комнате.

Пользователями веб-приложения могут быть частные предприниматели или государственные учреждения, заинтересованные в оптимизации процесса работы с посетителями, и непосредственно сами клиенты этих организаций.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	6
1 Анализ требований и уточнение спецификаций	7
1.1 Анализ задания и выбор технологии, языка и среды разработки	7
1.2 Выбор модели жизненного цикла программного обеспечения	8
1.3 Разработка диаграммы вариантов использования	8
1.4 Разработка концептуальной диаграммы классов.....	12
2 Проектирование структуры и компонентов программного продукта.....	14
2.1 Разработка структурной схемы программного продукта	14
2.2 Разработка интерфейса пользователя	15
2.2.1 Разработка иерархии меню.....	15
2.2.2 Построение диаграммы состояний интерфейса	16
2.2.3 Разработка форм интерфейса	18
2.3 Построение диаграммы классов уровня реализации.....	19
2.4 Проектирование базы данных	22
2.5 Разработка диаграммы компоновки.....	24
3 Выбор стратегии тестирования и разработка тестов	27
3.1 Структурный контроль.....	27
3.2 Модульное тестирование	28
3.3 Функциональное тестирование	30
3.4 Оценочное тестирование.....	32
ЗАКЛЮЧЕНИЕ.....	34
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	35
ПРИЛОЖЕНИЕ А. Техническое задание	36
ПРИЛОЖЕНИЕ Б. Руководство пользователя	45

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

БД – база данных

СУБД – система управления базами данных

ООП – объектно-ориентированное программирование

ТЗ – техническое задание

MVC (англ. model-view-controller) - схема разделения данных приложения, и управляющей логики на три отдельных компонента: модель, представление и контроллер.

Веб-приложение — клиент-серверное приложение, в котором клиент взаимодействует с веб-сервером при помощи браузера.

ORM (англ. object-relational mapping) — технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования.

IDE (англ. integrated development environment) – комплекс программных средств, используемый программистами для разработки программного обеспечения.

.NET – модульная платформа для разработки программного обеспечения с открытым исходным кодом.

ВВЕДЕНИЕ

Курсовая работа посвящена проектированию и разработке веб-приложения для организации очередей на обслуживание и расписаний приемов. Продукт может быть использован организациями, желающими отказаться от традиционной «живой» очереди или от записи на прием через регистратуру и перейти на онлайн-формат с минимальными затратами.

В настоящее время количество веб-приложений, позволяющих организовать онлайн-очередь или записаться на прием, довольно ограничено. В большинстве своем организации продолжают вести прием в порядке «живой» очереди или реализуют собственное интегрированное решение, которое не может быть использовано другими фирмами. Поэтому была выявлена необходимость в продукте, который мог бы быть использован повсеместно. Также организовать очередь и расписание приемов в удаленном формате в условиях пандемии кажется лучшим решением, чем живая очередь и очная запись. Это еще раз говорит о необходимости веб-приложения для организации очереди на обслуживание.

Таким образом, актуальность курсовой работы обусловлена, во-первых, тем, что она может быть использована любыми заинтересованными организациями, не зависимо от рода оказываемых услуг. Во-вторых, необходимостью подобного решения в условиях пандемии и, в-третьих, доступностью – для использования приложения необходим только доступ в Интернет.

1 Анализ требований и уточнение спецификаций

1.1 Анализ задания и выбор технологии, языка и среды разработки

Для разработки веб-приложения был выбран смешанный подход к программированию, объединяющий объектно-ориентированный подход и структурное программирование. Эта двойственность связана с тем, что предметная область программы – объектная и легко разбивается на отдельные взаимодействующие сущности, образующие иерархию наследования. В то же время для проектирования пользовательского интерфейса был выбран структурный подход, так как интерфейс представляет собой набор функций, обменивающихся данными.

Для реализации программного продукта был выбран свободно-распространяемый кроссплатформенный фреймворк ASP.NET Core для создания веб-приложений на платформе .NET, поддерживающий паттерн Model-View-Controller (MVC) [\[1\]](#). MVC предполагает разделение приложения на три компонента:

- модель – описание используемых в приложении данных, а также логики, которая связана непосредственно с данными;
- представление – отображение визуальной части или пользовательского интерфейса;
- контроллер – обеспечение связи между пользователем и приложением.

Описанный паттерн привлекателен тем, что все три компонента независимы, что позволяет реализовать концепцию разделения ответственности и облегчить разработку отдельных частей программы и их тестирование.

Для разработки серверной части приложения был выбран язык C#, как самый популярный из языков, поддерживаемый платформой .NET. Для реализации клиентской части использовался язык разметки HTML, формальный язык стилей CSS с подключенным набором инструментов Bootstrap и язык JavaScript совместно с библиотекой jQuery.

В качестве редактора исходного кода был выбран Visual Studio Code, позволяющий установить необходимые расширения как для работы с C#, так и для разработки представлений в веб-приложении, т.е. работы с HTML, CSS, JavaScript. Вместе с необходимыми расширениями VS Code позволяет использовать функции автодополнения кода, автоформатирования, подсказки параметров функций и обнаружение ошибок синтаксиса.

В качестве СУБД был выбран Microsoft SQL Server как надежная, производительная и простая в использовании СУБД. Microsoft SQL Server легко интегрируется в проекты на платформе .NET и имеет обширную и качественную документацию [\[2\]](#).

1.2 Выбор модели жизненного цикла программного обеспечения

Разработка веб-приложения часто сопровождается созданием прототипов. Прототип – это работающий программный продукт, который реализует отдельные функции и интерфейсы разрабатываемой системы.

В этой работе была использована спиральная модель жизненного цикла программного продукта, которая основана на прототипировании. Одно из главных преимуществ такой модели – возможность быстрого проектирования, разработки и тестирования первого рабочего прототипа, который можно использовать для презентации заказчику для уточнения требований.

Такой подход позволяет ускорить процесс формирования конечных требований заказчика за счет имеющейся практики использования программного продукта, уменьшить вероятность его морального устаревания за время разработки и облегчить процесс появления новых версий.

1.3 Разработка диаграммы вариантов использования

Проектирование веб-приложения начинается с определения основных вариантов его использования. В результате была разработана диаграмма вариантов использования или по-другому диаграмма прецедентов. Она позволяет описать взаимодействие различных пользователей с программным продуктом. В результате анализа ТЗ были получены следующие варианты взаимодействия веб-приложения и пользователя:

- регистрация;
- авторизация;
- изменение данных учетной записи;
- просмотр своих комнат;
- просмотр доступных комнат;
- создание комнаты;
- добавление пользователя в комнату;
- просмотр очередей и расписаний;
- постановка в очередь или запись на прием;
- сортировка и фильтрация отображенных элементов.

Для создателя комнаты доступны дополнительные варианты взаимодействия:

- назначение модераторов;
- удаление комнаты;
- создание очереди или расписания;

- изменение данных комнаты.

Основные варианты использования веб-приложения рассмотрены более подробно в таблицах 1-6.

Таблица 1 – Описание варианта использования «Создание комнаты»

Название варианта	Создание комнаты
Цель	Создать новую комнату
Действующие лица	Зарегистрированный пользователь
Краткое описание	Пользователь указывает данные комнаты и сохраняет ее
Тип	Основной

Таблица 2 – Вариант использования «Создание комнаты»

Действие пользователя	Отклик системы
1. Пользователь авторизуется.	2. Сервер обрабатывает запрос на авторизацию создает сессию, которую сохраняет в cookie.
3. Пользователь нажимает кнопку «Создать» на странице со списком комнат.	4. Сервер перенаправляет пользователя на страницу с формой для указания данных комнаты.
5. Пользователь заполняет поля и нажимает кнопку «Подтвердить».	6. Сервер валидирует данные и создает новую запись в таблице комнат в БД, после чего перенаправляет пользователя на страницу созданной комнаты.

Таблица 3 – Описание варианта использования «Создание очереди»

Название варианта	Создание очереди
Цель	Создать новую очередь
Действующие лица	Зарегистрированный пользователь
Краткое описание	Пользователь указывает данные очереди и сохраняет ее
Тип	Основной

Таблица 4 – Вариант использования «Создание комнаты»

Действие пользователя	Отклик системы
1. Пользователь авторизуется.	2. Сервер обрабатывает запрос на авторизацию создает сессию, которую сохраняет в cookie.
3. Пользователь выбирает одну из комнат на главной странице, после чего нажимает кнопку «Панель управления».	4. Сервер последовательно перенаправляет пользователя сначала на страницу комнаты, а затем на страницу управления комнатой.
5. Пользователь нажимает кнопку «Создать очередь» на странице управления комнатой.	6. Сервер перенаправляет пользователя на страницу с формой для указания данных очереди.
7. Пользователь заполняет поля и нажимает кнопку «Подтвердить».	8. Сервер валидирует данные и создает новую запись в таблице очередей в БД, после чего перенаправляет пользователя на страницу комнаты.

Таблица 5 – Описание варианта использования «Вступление в очередь»

Название варианта	Вступление в очередь
Цель	Присоединиться к существующей очереди
Действующие лица	Зарегистрированный пользователь
Краткое описание	Пользователь выбирает одну из доступных очередей и добавляет себя в список ее участников
Тип	Основной

Таблица 6 – Вариант использования «Вступление в очередь»

Действие пользователя	Отклик системы
1. Пользователь авторизуется.	2. Сервер обрабатывает запрос на авторизацию создает сессию, которую сохраняет в cookie.
3. Пользователь нажимает на кнопку «Очереди и расписания» на странице комнаты.	4. Сервер перенаправляет пользователя на страницу доступных очередей и расписаний.
5. Пользователь выбирает одну из очередей и нажимает кнопку «Присоединиться».	6. Браузер показывает пользователю модальное окно с подтверждением действия.
7. Пользователь подтверждает свой выбор.	8. Браузер скрывает модальное окно и заменяет кнопку «Присоединиться» на кнопку «Покинуть». В то же время сервер устанавливает связь между пользователем и выбранной очередью, создав новую запись в таблице участников очереди.

Для наглядного представления всех возможностей пользователя в рамках разрабатываемого веб-приложения была разработана диаграмма прецедентов, представленная на рисунке 1.



Рисунок 1 – Диаграмма прецедентов

Построенная диаграмма прецедентов позволила наглядно представить ожидаемое поведение системы, основных действующих лиц, варианты использования и связи – взаимодействие действующих лиц и соответствующих вариантов использования.

1.4 Разработка концептуальной диаграммы классов

Для определения более полной спецификации разрабатываемого программного обеспечения была разработана диаграмма классов концептуального уровня, представленная на рисунке 2. На этом уровне диаграмма классов демонстрирует связи между основными понятиями предметной области.

Ключевыми понятиями предметной области являются пользователь и комната. Комната агрегирует внутри себя такие понятия как очередь и расписание, причем в неограниченном количестве. При этом комнатой может владеть только один пользователь. Комната, пользователь, очередь и расписание описываются моделью, которая хранится в базе данных. Модель также используется сервисом доступа к базе данных для чтения, изменения и записи информации. Помимо сервиса доступа к базе данных в предметной области описываются сервис авторизации, определяющий роль текущего пользователя и

дающий ему соответствующие разрешения, и сервис отправки почты. Каждый из сервисов наследуется от базового сервиса с целью облегчения процесса интеграции в основную систему. Класс контроллера обрабатывает запросы и делает соответствующие обращения к сервису доступа к БД после чего формирует представление данных на основе полученного ответа.

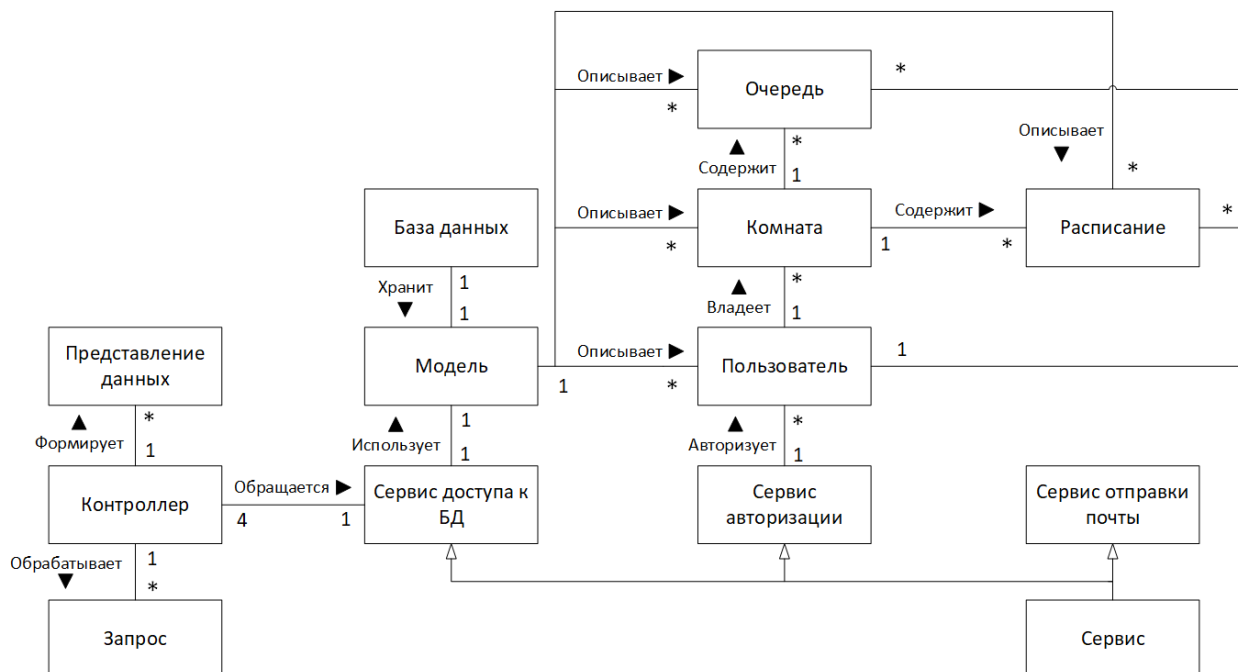


Рисунок 2 – Диаграмма классов концептуального уровня

Построенная диаграмма классов концептуального уровня обладает значительной степенью абстракции, поэтому не все выделенные классы останутся классами на уровне реализации. Однако эта схема является центральным звеном объектно-ориентированного метода разработки программного обеспечения, поэтому дальнейшие архитектурные решения будут во многом основываться именно на ней.

2 Проектирование структуры и компонентов программного продукта

2.1 Разработка структурной схемы программного продукта

Так как для разработки веб-приложения использовался паттерн MVC, который предполагает разделение приложения на три основных компонента, в программном продукте были выделены три подсистемы – подсистема контроллеров (собственно, контроллер в паттерне MVC), подсистема отображения информации (представление) и подсистема управления доступом к БД (модель).

Подсистема отображения информации отвечает за состояние пользовательского интерфейса. Эта подсистема включает в себя непосредственно представления – шаблонизированные HTML-страницы, которые сервер передает клиенту по запросу, стили – описание внешнего вида HTML-страниц, выполненное с помощью языка CSS, и модели представлений, которые используются для передачи данных в представления.

Подсистема контроллеров включает четыре контроллера, каждый из которых отвечает за определенную область работы программы: контроллер пользователя (регистрация, авторизация и так далее), контроллер комнаты (создание, удаление и так далее), контроллер очереди (добавление участников, контроль продвижения и так далее) и контроллер расписаний (назначение приемов, запись на прием и так далее).

С целью уменьшения зависимости контроллеров от моделей все взаимодействие с базой данных было вынесено в отдельную подсистему – подсистему управления доступом к БД. Эта подсистема отвечает за любые запросы к базе данных, которые надо выполнить серверу. В качестве ORM решения был использован ADO.NET Entity Framework, который требует определения контекста данных для доступа к базе данных и выполнения запросов [3]. Поэтому в подсистему управления доступом к БД помимо самих моделей был включен контекст данных.

Также веб-приложение обладает набором внутренних сервисов, то есть сервисов, которые не взаимодействуют с пользователем напрямую. Это сервис авторизации и контроля доступа, определяющий роль текущего пользователя и дающий ему разрешения на те или иные действия, сервис контроля фоновых задач, связанный в первую очередь с отправкой уведомлений в определенное время, и сервис рассылки электронных писем. Все эти сервисы объединены в подсистему внутренних сервисов.

Последняя подсистема называется подсистемой обработчиков событий. Она разделена на две части – обработчики глобальных событий, к которым в свою очередь относятся обработчик уведомлений, обработчик изменения состояния очереди и обработчик чата, и обработчик локальных событий. К локальным событиям относятся

события, затрагивающие только интерфейс одного пользователя – отображение модального окна по нажатию кнопки, например.

Итоговая структурная схема веб-приложения представлена на рисунке 4.

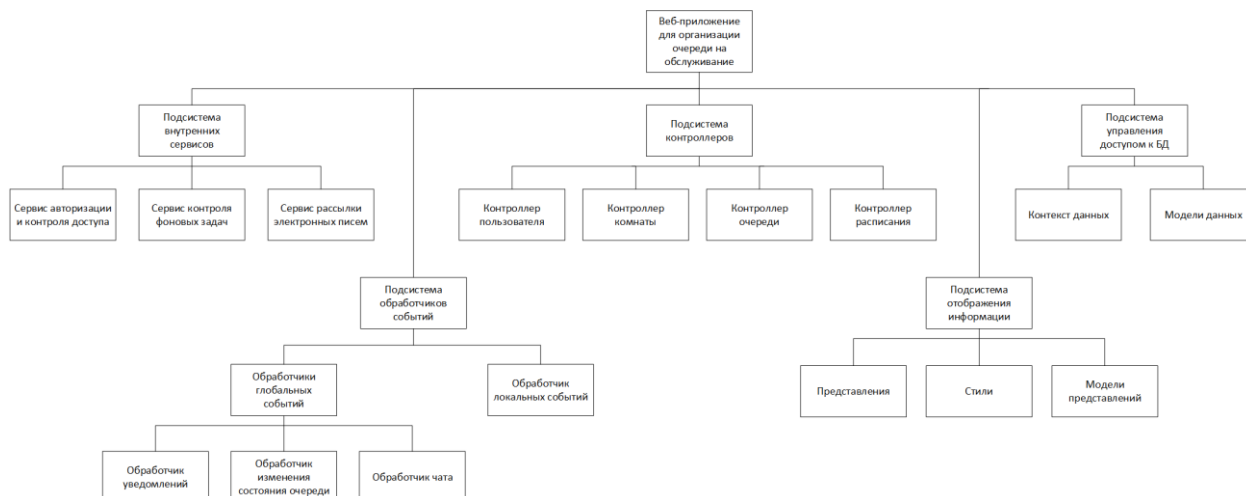


Рисунок 4 - Структурная схема веб-приложения

Структурная схема веб-приложения для организации очереди на обслуживание содержит 5 подсистем.

2.2 Разработка интерфейса пользователя

2.2.1 Разработка иерархии меню

Принимая во внимание особенности каждой из сущностей, присутствующих на концептуальном уровне программного продукта (пользователь, комната, очередь и расписание), можно сделать вывод, что создание единого унифицированного меню крайне затруднительно. В первую очередь из-за того, что при попытке его создания, интерфейс получится перегруженным и, как следствие, непригодным для нормального использования.

Тем не менее, некоторые элементы, которые желательно иметь в быстром доступе на любом этапе работы веб-приложения, выделить можно. Сюда относится переход на главную страницу, являющуюся «отправной точкой» всего интерфейса, отображение списка уведомлений и пользовательская панель – для удобного доступа к любой информации о текущем пользователе.

Сделанные выводы позволили разработать схему иерархии меню верхней панели веб-приложения, где будут расположены ключевые элементы, описанные выше. Иерархия меню представлена на рисунке 5.



Рисунок 5 – Иерархии меню

Разработанные иерархии меню позволяют получить представление о форме и уровне вложенности меню верхней панели.

2.2.2 Построение диаграммы состояний интерфейса

При проектировании интерфейса были приняты во внимание следующие правила с целью сделать интерфейс понятным и простым для пользователя:

- интерфейс должен быть последовательным: все элементы должны быть уместными, а ответ на действия пользователя – предсказуемым;
- должны присутствовать определённые паттерны: например, наличие навигационной панели в верхней части экрана;
- интерфейс не должен быть перегружен: на экране должны присутствовать такое количество элементов, чтобы нужное действие можно было найти без лишних усилий;
- должна присутствовать возможность вернуться назад: отменить последнее действие или вернуться на главную страницу;
- должно быть предусмотрено разделение доступа: неавторизованные пользователи не имеют доступа к основному интерфейсу, обычные пользователи не имеют доступа к панелям управления и так далее.

С учетом всех этих правил, других требований к пользовательскому интерфейсу и проектных решений, принятых выше, была построена диаграмма состояний интерфейса, представленная на рисунке 6.

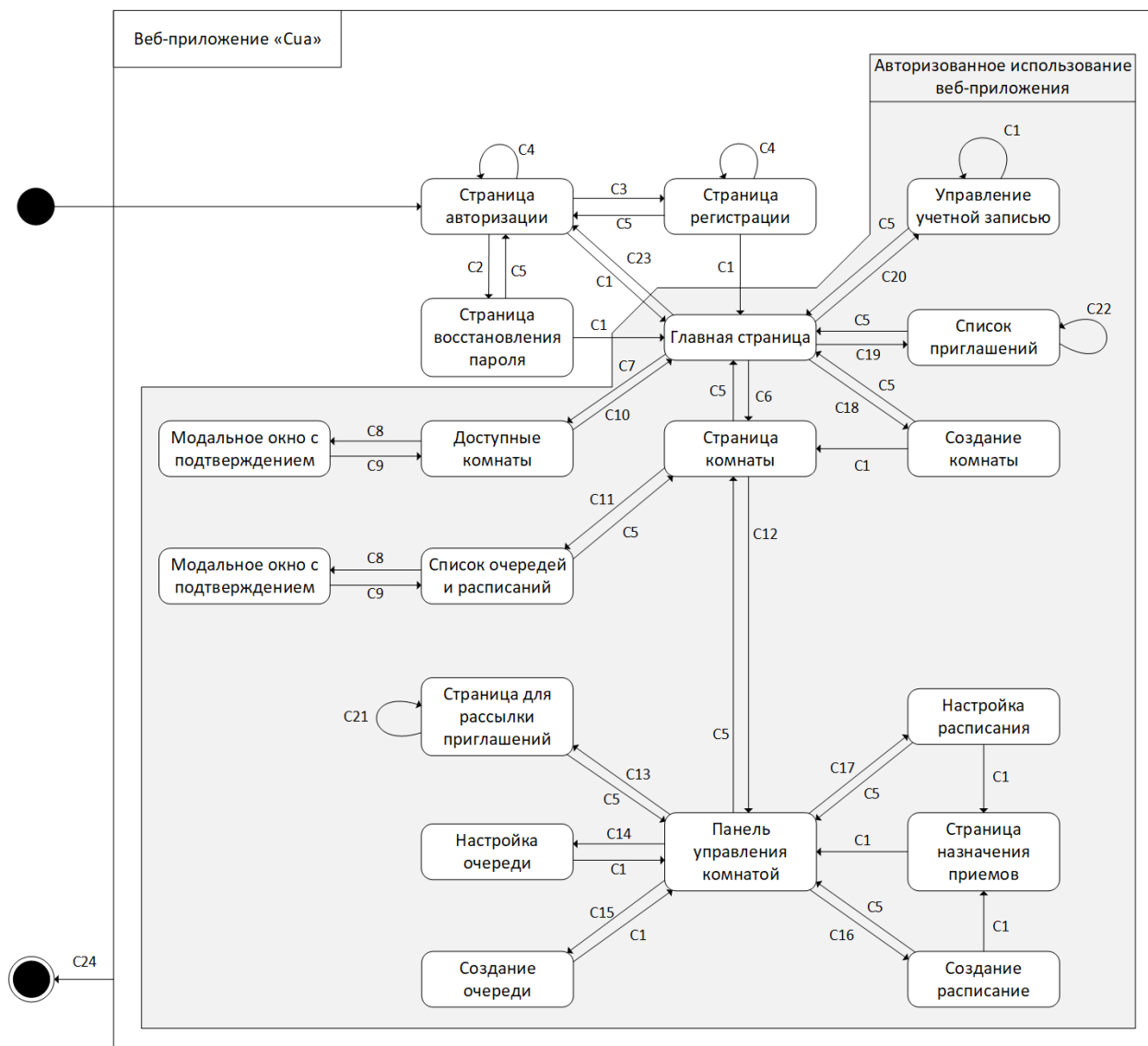


Рисунок 6 – Диаграмма состояний интерфейса

Принятые обозначения:

- C1 – ввод корректных данных и подтверждение;
- C2 – нажатие кнопки «Восстановить пароль»;
- C3 – нажатие кнопки «Регистрация»;
- C4 – ввод некорректных данных;
- C5 – нажатие кнопки «Назад»;
- C6 – выбор одной из комнат;
- C7 – нажатие кнопки «Присоединиться»;
- C8 – выбор одного из доступных вариантов;
- C9 – подтверждение выбора;
- C10 – нажатие на иконку «Cua»;
- C11 – нажатие кнопки «Список очередей и расписаний»;
- C12 – нажатие кнопки «Панель управления»;

- C13 – нажатие кнопки «Добавить участника»;
- C14 – нажатие на иконку шестеренки рядом с названием очереди;
- C15 – нажатие кнопки «Создать очередь»;
- C16 – нажатие кнопки «Создать расписание»;
- C17 – нажатие на иконку шестеренки рядом с названием расписания;
- C18 – нажатие кнопки «Создать комнату»;
- C19 – переход по ссылке «Список приглашений»;
- C20 – переход по ссылке «Настройка аккаунта»;
- C21 – отправка приглашения пользователю;
- C22 – принятие или отклонение приглашения;
- C23 – переход по ссылке «Выйти»;
- C24 – закрытие страницы веб-приложения.

Разработанная диаграмма состояний интерфейса позволила уточнить возможные действия пользователя в рамках взаимодействия с интерфейсом веб-приложения.

2.2.3 Разработка форм интерфейса

Веб-приложение имеет 19 представлений, которые в совокупности представляют пользовательский интерфейс. Среди представлений можно выделить основные, то есть те, с которыми пользователь взаимодействует большую часть времени, и дополнительные, которые используются для выполнения специализированных действий.

К основным представлениям (веб-страницам) относятся:

- страница авторизации;
- страница регистрации;
- главная страница;
- страница со списком доступных комнат;
- страница комнаты;
- страница со списком очередей и расписаний;
- страница управления комнатой.

Между страницами возможно переключение с помощью специальных навигационных элементов, расположенных как на навигационной панели в верхней части экрана, так и на самой странице. Все страницы имеют схожую компоновку: главная информация находится по центру и занимает большую часть экрана, вспомогательная – по бокам. Каждый компонент на странице выполнен в виде отдельного блока с информацией. Элементы, с которыми можно взаимодействовать, выделены синим или красным цветом в зависимости от предполагаемого результата.

В качестве примера на рисунках 7 и 8 приведено подробное описание форм интерфейса «Страница комнаты» и «Страница со списком очередей и расписаний».

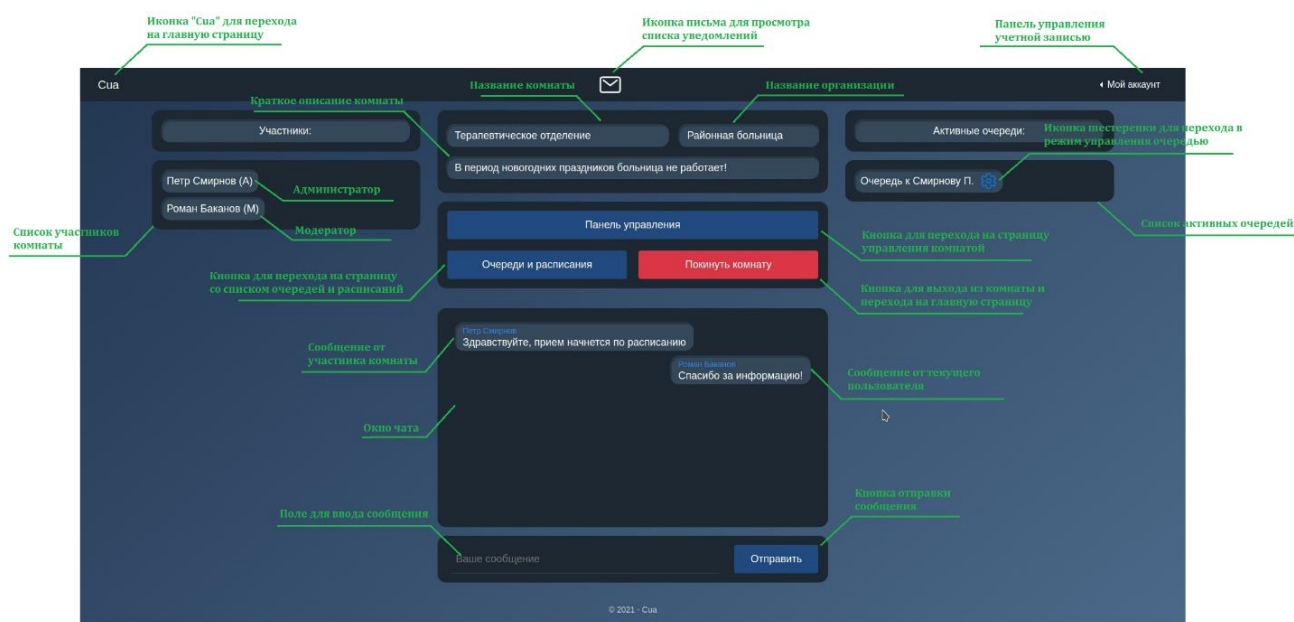


Рисунок 7 – Страница комнаты



Рисунок 8 – Страница со списком очередей и расписаний

2.3 Построение диаграммы классов уровня реализации

Так как для разработки серверной части был выбран объектный подход, в ходе работы была построена диаграмма классов уровня реализации [5], представленная на рисунке 9.

Классы контроллеров AccountController, RoomController, QueueController и TimetableController наследуются от системного класса Controller, предоставляющего базовый функционал контроллера паттерна MVC. Они также связаны отношением композиции с такими классами как AuthorizationService, MailService и DBAccessService для

доступа к методам внутренних сервисов. Класс MailService связан отношением композиции с классом MailSettings, он использует этот класс для настройки почтового клиента. Класс DBAccessService отвечает за взаимодействие с базой данных, поэтому связан отношением композиции с классом ApplicationContext, который предоставляет доступ к базе данных. ApplicationContext наследуется от системного класса DbContext, отражающего состояние базы данных, и связан отношением композиции с системным классом DbSet, который предоставляет доступ к одной из таблиц в базе данных. Также DBAccessService агрегирует внутри себя класс HubContext<MainHub>, полученный связыванием из параметризованного класса HubContext.

2.4 Проектирование базы данных

Разрабатываемый программный продукт работает с учетными данными пользователей, созданными ими комнатами, очередями и расписаниями. Также необходимо хранить информацию об участниках комнат, очередей и расписаний. К дополнительной информации относятся заявки на вступление в комнату, уведомления и сообщения чата.

На рисунке 10 изображена схема спроектированной базы данных.

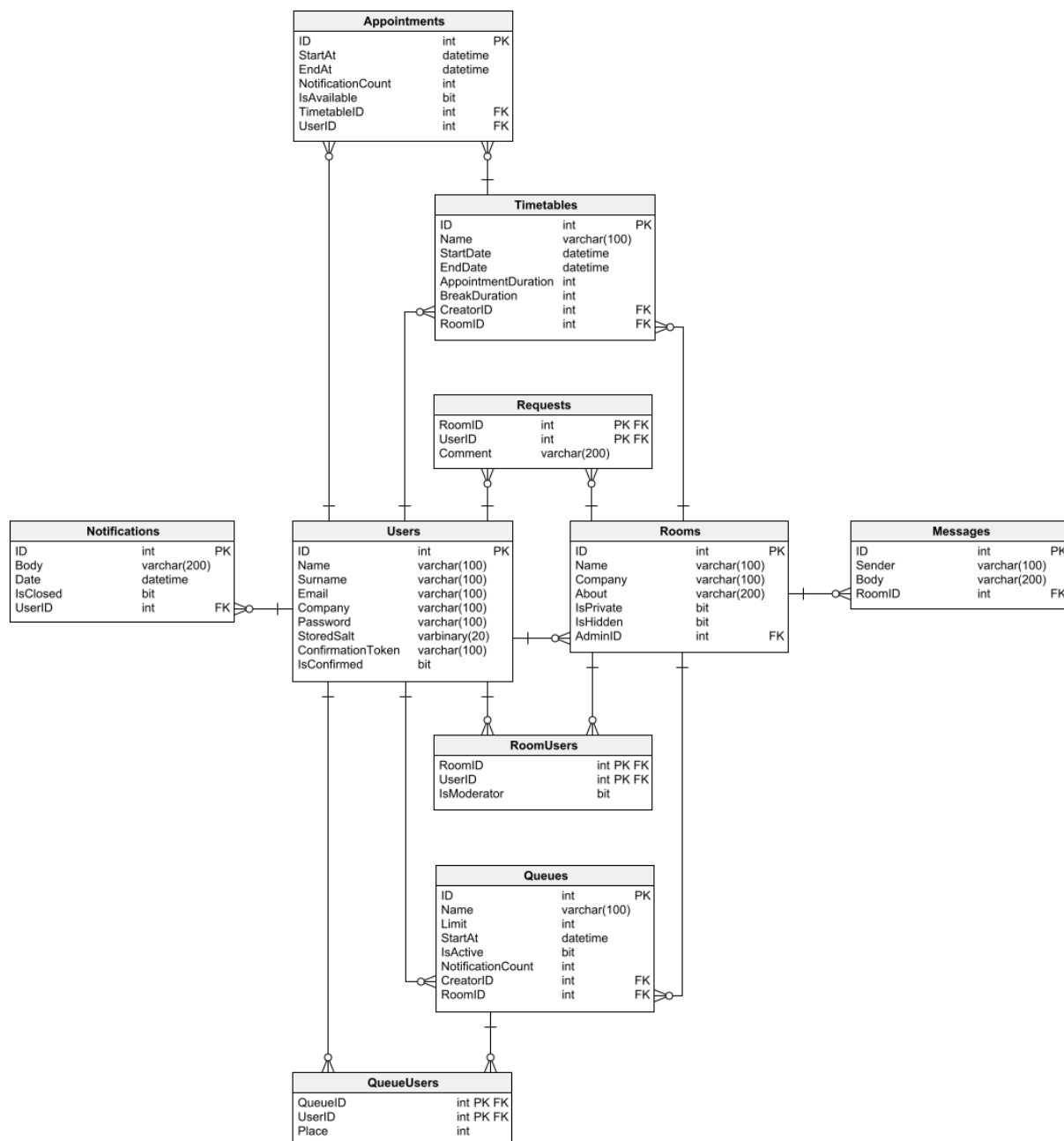


Рисунок 10 – Схема базы данных

В базе данных присутствуют следующие таблицы и поля:

- Users – учетные данные пользователей:

- 1) ID – уникальный идентификатор пользователя;
 - 2) Name – имя пользователя;
 - 3) Surname – фамилия пользователя;
 - 4) Email – электронная почта пользователя;
 - 5) Company – организация пользователя;
 - 6) Password – пароль пользователя;
 - 7) StoredSalt – соль для хеширования пароля;
 - 8) ConfirmationToken – токен для верификации пользователя;
 - 9) IsConfirmed – информация о состоянии верификации пользователя;
- Rooms – данные комнат:
 - 1) ID – уникальный идентификатор комнаты;
 - 2) Name – название комнаты;
 - 3) Company – организация, владеющая комнатой;
 - 4) About – краткая информация о комнате;
 - 5) IsPrivate – информация о том, является ли комната частной;
 - 6) IsHidden – информация о том, является ли комната скрытой;
 - Queues – данные очередей:
 - 1) ID – уникальный идентификатор очереди;
 - 2) Name – название очереди;
 - 3) Limit – ограничение на количество людей в очереди;
 - 4) StartAt – дата начала приема очереди;
 - 5) IsActive – информация о том, идет ли прием очереди;
 - Timetables – данные расписаний:
 - 1) ID – уникальный идентификатор расписания;
 - 2) Name – название расписания;
 - 3) StartDate – дата первого приема по расписанию;
 - 4) EndDate – дата последнего приема по расписанию;
 - 5) AppointmentDuration – длительность приема;
 - 6) BreakDuration – длительность перерыва;
 - Requests – таблица приглашений и заявок:
 - 1) Comment – комментарий к заявке или приглашению;
 - 2) RoomID – поле для связи с таблицей Rooms;
 - 3) UserID – поле для связи с таблицей Users;
 - QueueUsers – таблица участников очереди:
 - 1) Place – место человека в очереди;

- 2) QueueID – поле для связи с таблицей Queues;
- 3) UserID – поле для связи с таблицей Users;
- Appointments – данные о приемах:
 - 1) ID – уникальный идентификатор приема;
 - 2) StartAt – время начала приема;
 - 3) EndAt – время окончания прием;
 - 4) IsAvailable – информация, доступен ли прием для записи.
- RoomUsers – таблица участников комнаты:
 - 1) IsModerator – информация о том, является ли пользователь модератором;
 - 2) RoomID – поле для связи с таблицей Rooms;
 - 3) UserID – поле для связи с таблицей Users;
- Messages – данные сообщений:
 - 1) ID – уникальный идентификатор сообщения;
 - 2) Sender – имя отправителя;
 - 3) Body – текст сообщения;
- Notifications – данные уведомлений:
 - 1) ID – уникальный идентификатор уведомления;
 - 2) Body – текст уведомления;
 - 3) Date – дата уведомления;
 - 4) IsClosed – информация о том, скрыто ли уведомление.

2.5 Разработка диаграммы компоновки

Диаграмма компоновки отражает физическую структуру разрабатываемого программного продукта. На данной диаграмме показано как выглядит приложение на программном уровне и из каких частей оно состоит, стрелками обозначены зависимости компонентов и модулей приложения.

На диаграмме компонентов веб-приложения, изображенной на рисунке 11, отражены программные компоненты и файлы, которые выделены в отдельные модули, эквивалентные директориями операционной системы. Веб-приложение состоит из 7 модулей. Первый и самый главный модуль Application является корневой директорией всего приложения, в нем определены ключевые программные компоненты, запускающие работу веб-приложения и определяющие его конфигурацию: набор сервисов, маршрутизацию и так далее. В модуле Controllers хранятся контроллеры, являющиеся связующим компонентом между представлениями и сервисом, работающим с моделями. Сами представления хранятся в модуле Views, они структурированы в подмодули в

соответствии с содержанием отображаемой информации. В модуле `wwwroot` содержатся файлы стилей, файлы JavaScript и другие статические элементы, которые используются на стороне клиента. Модуль `Services` объединяет внутренние сервисы, в частности – сервис взаимодействия с базой данных, который связан с контекстом из модуля `Models`. Помимо контекста в модуле `Models` хранятся модели, которые Entity Framework использует для осуществления связи с БД. В качестве источника данных обозначен Microsoft SQL Server, взаимодействие с которым происходит с помощью упомянутого выше Entity Framework.

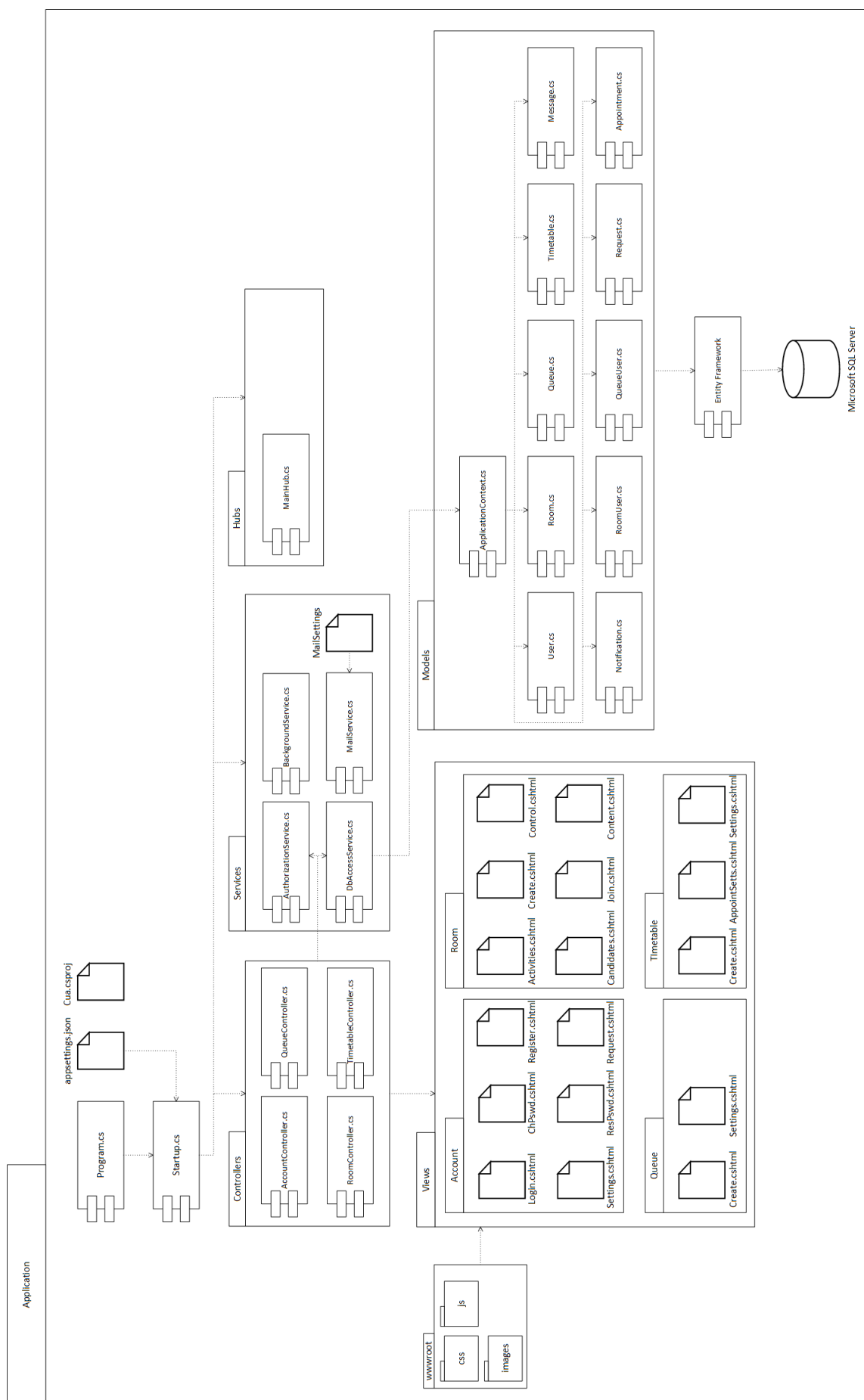


Рисунок 11 – Диаграмма компоновки

3 Выбор стратегии тестирования и разработка тестов

3.1 Структурный контроль

В качестве одного из методов тестирования был выбран структурный контроль с целью обеспечения как можно более раннего обнаружения ошибок в процессе разработки. Использование структурного контроля на заранее определенных контрольных точках помогает обнаружить типовые ошибки, наиболее часто встречающиеся в ходе разработки программного продукта. Этапы и результаты заключительного структурного контроля приведены в таблице 7.

Таблица 7 – Структурный контроль

Вопрос	Результат структурного контроля	Вывод
Все ли переменные инициализированы?	Да, иначе IDE показала бы ошибку.	Все переменные инициализированы.
Присутствуют ли переменные со сходными именами?	Все переменные имеют различные (и различимые) имена.	Переменные со сходными именами отсутствуют.
Не выходят ли индексы за границы массива?	Индексы не выходят за границы массива, так как перебор происходит в рамках специальных итерационных циклов.	Индексы не выходят за границы массива.
Будут ли корректно завершены циклы?	Все итерационные циклы будут корректно завершены.	Циклы будут корректно завершены.
Будет ли завершена программа?	Каждый запрос к серверу получит ответ, содержащий или желаемую информацию, или сообщение об ошибке.	Программа будет завершена.
Существуют ли поисковые циклы? Корректно ли отрабатываются ситуации «элемент найден» и «элемент не найден»?	Поисковых циклов нет.	Поисковых циклов нет.

Продолжение таблицы 7

Соответствуют ли списки параметров и аргументов методов по порядку, типу, единицам измерения?	Да, иначе IDE показала бы ошибку.	Списки параметров и аргументов полностью совпадают.
Не изменяет ли метод аргументов, которые не должны изменяться?	Не изменяет, все неизменяемые аргументы помечены модификатором «in».	Метод не изменяет аргументов, которые не должны изменяться.
Не происходит ли нарушения области действия глобальных и локальных переменных с одинаковыми результатами?	Переменных с одинаковым названием нет.	Нарушения области действия глобальных и локальных переменных не происходит.
Использованы ли нетипизированные переменные, открытые массивы, динамическая память? Если да, то соответствуют ли типы переменных при «наложении» формата?	Использованы нетипизированные переменные и динамические массивы. Все приведения типов выполнены корректно.	Обращение к нетипизированным переменным не вызывает ошибок.
Корректно ли выполнены вычисления с переменными различных типов (в том числе с использованием целочисленной арифметики)?	Да, иначе IDE показала бы ошибку.	Вычисления переменных различных типов выполнены корректно.

Анализ результатов структурного контроля позволяет сделать вывод, что итоговая версия продукта не обладает типовыми ошибками кодирования.

3.2 Модульное тестирование

Вторым методом тестирования было выбрано модульное тестирование. Цель модульного тестирования – изолировать отдельные части программы и проверить, что по

отдельности эти части работоспособны. Данный метод тестирования предполагает написание теста для каждого нетривиального метода и позволяет достаточно быстро проверить, не привело ли очередное изменение кода к появлению ошибок в уже оттестированных местах программного продукта. В ходе тестирования были написаны модульные тесты для большей части методов. При этом тестировалось не только корректное выполнение участка кода, но и ситуации, которые должны генерировать ошибки.

Для имитации работы нижних слоев архитектуры применялись моки (mock) – объекты-заглушки, генерируемые внутри теста и конфигурируемые под конкретное поведение. Вне теста моки использоваться не могут. Для интеграции модульных тестов в проект использовался набор инструментов xUnit, а для создания моков – фреймворк Moq.

Примеры модульных тестов, их описание, результаты и вывод, сделанный по итогу их выполнения, представлены в таблице 8.

Таблица 8 – Модульное тестирование

Название теста	Описание теста	Ожидаемый результат	Полученный результат	Вывод
IndexModelTest	Проверка того, что метод Index контроллера комнаты передает в представление корректный список комнат.	Передается список комнат, созданных пользователем или добавленных через кнопку «Присоединиться».	Передается список комнат, созданных пользователем или добавленных через кнопку «Присоединиться».	Список комнат формируется корректно.
CreateRoomView Test	Проверка того, что типом возвращаемого значения метода Create контроллера комнаты является представление.	По get-запросу возвращается представление.	По get-запросу возвращается представление.	Тип возвращаемого значения является корректным.
UpdateRoomTest	Проверка того, что метод UpdateRoom сервиса доступа к базе данных обновляет нужную запись в базе данных.	Соответствующая запись в базе данных обновляется.	Соответствующая запись в базе данных обновляется.	Метод корректно взаимодействует с базой данных.

Продолжение таблицы 8

DeleteQueueTest	Проверка того, что метод DeleteQueue сервиса доступа к базе данных удаляет нужную запись в базе данных.	Соответствующая запись в базе данных удаляется.	Соответствующая запись в базе данных удаляется.	Метод корректно взаимодействует с базой данных.
AdminDetection Test	Проверка того, что метод IsAdmin сервиса авторизации корректно определяет роль пользователя.	Возвращается значение true.	Возвращается значение true	Метод корректно определил, что пользователь является администратором.

Анализ результатов модульного тестирования позволяет сделать вывод, что отдельные компоненты веб-приложения работают корректно.

3.3 Функциональное тестирование

Следующим этапом тестирования стало функциональное тестирование. Данный вид тестирования предполагает, что неизвестно, как работает программа, однако при этом важность имеет корректность обработки входных данных и получение ожидаемого результата. То есть, функциональное тестирование определяет способность системы в заданных условиях правильно отвечать на запросы пользователей.

Было принято решение воспользоваться методом причинно-следственных связей, так как он позволяет системно выбирать высокорезультативные тесты. Метод оперирует понятиями «причина», то есть отдельное входное условие, и «следствие» - выходное условие или преобразование системы [4]. Идея метода заключается в отнесении всех следствий к причинам, то есть в установлении соответствия между входными и выходными данными.

Для автоматизации процесса функционального тестирования была использована библиотека Selenium. Со списком функциональных тестов методом анализа причинно-следственных связей можно ознакомиться в таблице 9.

Таблица 9 – Тестирование методом анализа причинно-следственных связей

Тест	Ожидаемый результат	Полученный результат	Вывод
Попытка авторизации с некорректными данными.	Вывод сообщения о некорректно введенных данных.	Вывод сообщения о некорректно введенных данных.	Авторизация с некорректными данными невозможна.
Попытка авторизации с корректными данными.	Переадресация на главную страницу.	Переадресация на главную страницу.	Авторизация работает корректно.
Создание комнаты.	Переадресация на главную страницу, появление новой комнаты на главной странице.	Переадресация на главную страницу, появление новой комнаты на главной странице.	Создание комнаты работает корректно.
Удаление комнаты.	Переадресация на главную страницу, исчезновение комнаты с главной страницы.	Переадресация на главную страницу, исчезновение комнаты с главной страницы.	Удаление комнаты работает корректно.
Создание очереди.	Переадресация на страницу управления комнатой, появление новой очереди в списке созданных очередей и расписаний.	Переадресация на страницу управления комнатой, появление новой очереди в списке созданных очередей и расписаний.	Очередь создается корректно.
Создание расписания.	Переадресация на страницу управления комнатой, появление нового расписания в списке созданных очередей и расписаний.	Переадресация на страницу управления комнатой, появление нового расписания в списке созданных очередей и расписаний.	Расписание создается корректно.
Удаление очереди.	Исчезновение очереди из списка созданных очередей и расписаний.	Исчезновение очереди из списка созданных очередей и расписаний.	Удаление очереди работает правильно.

Продолжение таблицы 9

Удаление расписания.	Исчезновение расписания из списка созданных очередей и расписаний.	Исчезновение расписания из списка созданных очередей и расписаний.	Удаление расписания работает правильно.
Постановка пользователя в очередь	Исчезновение кнопки «Присоединиться» и появление кнопки «Покинуть» на карточке очереди.	Кнопка «Присоединиться» не исчезает, а кнопка «Покинуть» не появляется.	Процесс постановки пользователя в очередь работает с ошибкой.
Запись на прием.	Исчезновение кнопки «Записаться» и появление кнопки «Отменить запись» на карточке расписания.	Исчезновение кнопки «Записаться» и появление кнопки «Отменить запись» на карточке расписания.	Пользователь может записаться на прием.
Исключение пользователя из очереди.	Исчезновение кнопки «Покинуть» и появление кнопки «Присоединиться» на карточке очереди.	Исчезновение кнопки «Покинуть» и появление кнопки «Присоединиться» на карточке очереди.	Исключение пользователя из очереди работает корректно.
Отмена записи на прием.	Исчезновение кнопки «Отменить запись» и появление кнопки «Записаться» на карточке расписания	Появление ошибки, связанной с указателем на значение null.	Процесс отмены записи на прием работает с ошибкой.

Анализ результатов функционального тестирования позволил выявить две ошибки и исправить их. В остальном можно сделать вывод, что поведение системы соответствует норме, и на любой запрос пользователя дается правильный и ожидаемый ответ.

3.4 Оценочное тестирование

Оценочное тестирование применялось на заключительном этапе испытаний программного продукта. Целью оценочного тестирования является получение информации о том, соответствует ли программа основным предъявленным требованиям. Этот вид тестирования включает в себя множество этапов, начиная с тестирования производительности и заканчивая тестированием удобства установки. Однако на практике обычно выполняются не все этапы оценочного тестирования, так как это очень дорого и трудоемко. С учетом особенностей разработанного веб-приложения были определены

следующие важные этапы, которые система должна проходить в процессе оценочного тестирования:

- тестирование удобства использования – проверка соответствия программного продукта основным положениям технического задания (данный вид тестирования проводился совместно с пользователями, не знакомыми с интерфейсом программного продукта, – им предлагалось оценить удобство использования веб-приложения при выполнении основных функций, все опрошенные дали положительную оценку удобства использования);
- тестирование удобства эксплуатации – анализ психологических факторов, возникающих при работе с программным продуктом и, как следствие, проверка удобства интерфейса (данный вид тестирования проводился совместно с пользователями, не знакомыми с интерфейсом программного продукта, – им предлагалось дать ответ на вопрос «Не вызывает ли интерфейс утомления, дискомфорта и иных негативных воздействий?», все респонденты ответили на поставленный вопрос отрицательно);
- тестирование защиты – проверка невозможности несанкционированного доступа к информации;
- тестирование совместимости – проверка корректности работы программного продукта в разных средах (в случае веб-приложения – в разных браузерах);
- тестирование восстановления – проверка восстановления программного обеспечения после сбоев (например, некорректного завершения работы);
- тестирование многопользовательского режима – проверка возможности работы с программным продуктом несколькими пользователями одновременно;
- тестирование на больших объемах – проверка работоспособности программного продукта на больших объемах данных;
- тестирование режима реального времени – проверка корректности работы программного продукта при переходе в режим реального времени.

В результате прохождения веб-приложением всех этапов оценочного тестирования, можно сделать вывод, что оно в полном объеме удовлетворяет требованиям к функционированию системы, указанным в ТЗ.

ЗАКЛЮЧЕНИЕ

В результате выполнения курсовой работы было спроектировано и реализовано веб-приложение, удовлетворяющая всем требованиям технического задания. Программный продукт предоставляет возможность создания комнат, очередей и расписаний, вступления в них и контроля изменения их состояния. Также реализованы системы уведомлений, рассылки электронных писем и ограничения доступа к комнатам.

Система может быть расширена в последующих версиях. В качестве усовершенствования рассматривается добавление возможности вступать в комнаты и очереди по QR-коду, введение роли администратора и системы жалоб.

В ходе разработки веб-приложения был приобретён опыт проектирования программных систем, работы с системой контроля версий, работы с БД и фреймворком ASP.NET Core, а также получены навыки составления технической документации.

Проект расположен в репозитории по адресу:

<https://github.com/Rembler/BMSTU-Cua>

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ASP.NET Core documentation | Microsoft Docs [Электронный ресурс]. – URL: <https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-6.0> (Дата обращения: 15.10.2021).
2. SQL Server technical documentation - SQL Server | Microsoft Docs [Электронный ресурс]. – URL: <https://docs.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver15> (Дата обращения: 20.10.2021).
3. Entity Framework documentation | Microsoft Docs [Электронный ресурс]. – URL: <https://docs.microsoft.com/en-us/ef/> (Дата обращения: 20.10.2021).
4. Иванова Г.С. – Технология программирования: учебник / Г.С. Иванова – 3-е изд., стер. – М.: КНОРУС, 2016. – 334 с. – (Бакалавриат).
5. Иванова Г.С., Ничушкина Т.Н., Пугачев Е.К., Самарев Р.С., Фетисов М.В. – Методические указания по выполнению курсовой работы по дисциплине «Технология разработки программных систем».: Электронное учебное издание. – МГТУ им. Н.Э. Баумана, 2019. – 41 с.

ПРИЛОЖЕНИЕ А
Техническое задание

**Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)**

**ФАКУЛЬТЕТ ИНФОРМАТИКА, ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ И СИСТЕМЫ
УПРАВЛЕНИЯ**

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.01 Информатика и вычислительная техника

ВЕБ-ПРИЛОЖЕНИЕ «СUA» ДЛЯ ОРГАНИЗАЦИИ ОЧЕРЕДИ НА ОБСЛУЖИВАНИЕ

**Техническое задание на курсовую работу
по дисциплине Технология разработки программных систем**

Листов 8

Студент ИУ6-52Б
(Группа)

ИУ6 20.09.2021
(Подпись, дата)

Р.В. Баканов
(И.О. Фамилия)

Руководитель курсовой работы,
(д.т.н., проф.)

20.09.2021
(Подпись, дата)

Г.С. Иванова
(И.О. Фамилия)

Москва, 2021

1 ВВЕДЕНИЕ

Настоящее техническое задание распространяется на разработку веб-приложения «Cua», используемого для организации очередей и онлайн-записи на обслуживание.

Актуальность работы обусловлена тем, что многие компании из сферы услуг до сих пор ведут запись клиентов по телефону и заполняют бумажный журнал. Нередки также случаи, когда прием ведется в порядке живой очереди. Переход же на онлайн-формат избавит работника от лишней бумажной работы, а посетителя – от необходимости длительное время стоять в живой очереди. Актуальность реализации обусловлена тем, что компании, желающей использовать онлайн-очереди и онлайн-запись в своей деятельности, не требуется разрабатывать собственное программное обеспечение для этого, а клиенту – что-либо устанавливать на свое устройство, так как для использования веб-приложения «Cua» требуется только доступ в Интернет.

2 ОСНОВАНИЯ ДЛЯ РАЗРАБОТКИ

Веб-приложение «Cua» разрабатывается по личной инициативе автора.

3 НАЗНАЧЕНИЕ РАЗРАБОТКИ

Основное назначение веб-приложения «Cua» заключается в возможности создания пользователем отдельных комнат, а также очередей и расписаний внутри этих комнат. Другим пользователям соответственно предоставляется возможность присоединяться к созданным комнатам, записываться в очереди и на прием. Также пользователи могут регистрироваться, а также редактировать и искать комнаты.

4 ТРЕБОВАНИЯ К ПРОГРАММНОМУ ИЗДЕЛИЮ

4.1 Требования к функциональным характеристикам

4.1.1 Выполняемые функции

4.1.1.1 Для незарегистрированного пользователя:

- регистрация;

4.1.1.2 Для зарегистрированного пользователя:

- авторизация посредством логина и пароля;
- изменение данных учетной записи;
- удаление учетной записи;
- создание комнат;
- поиск комнат;
- вступление в комнату.
- вступление в очередь;
- запись на прием;
- удаление из очереди;
- отмена записи в расписании;
- выход из комнаты;
- отправка сообщений в чат комнаты.

4.1.1.3 Для создателя комнаты:

- изменение параметров комнаты;
- удаление комнаты;
- создание очередей и расписаний;
- изменение параметров очередей и расписаний;
- удаление очередей и расписаний;
- принятие и отклонение заявок на вступление в комнату;
- исключение участников;
- контроль продвижения очереди.

4.1.2 Исходные данные:

4.1.2.1 Авторизационные данные:

- адрес электронной почты;
- пароль.

4.1.2.2 Регистрационные данные:

- имя;
- пол;
- адрес электронной почты;
- пароль;
- номер телефона;
- место работы / учебы.

4.1.2.3 Данные комнаты:

- название;
- краткое описание;
- приватность;
- видимость.

4.1.2.4 Данные очереди:

- название;
- ограничение на количество участников;
- время начала приема;

4.1.2.5 Данные расписания:

- название;
- даты ведения приема;
- время ведения приема;
- время, отводимое на один прием;

4.2 Требования к надежности

4.2.1 Предусмотреть контроль вводимой информации.

4.2.2 Предусмотреть защиту от некорректных действий пользователя.

4.2.3 Обеспечить целостность информации в базе данных.

4.2.4 Обеспечить корректную работу системы в режиме реального времени.

4.3 Условия эксплуатации

4.3.1 Условия эксплуатации в соответствии с СанПиН 2.2.2/2.4.1340-03.

4.4 Требования к составу и параметрам технических средств

4.4.1 Программное обеспечение должно функционировать на IBM-совместимых персональных компьютерах.

4.4.2 Минимальная конфигурация технических средств:

4.4.2.1 Тип процессора Intel Pentium 4 или более поздней версии.

4.4.2.2 Объем ОЗУ 1 Гб.

4.4.2.3 Свободное место на диске 1 Гб.

4.5 Требования к информационной и программной совместимости

4.5.1 Веб-приложение должно корректно работать с веб-браузерами, начиная с версий Google Chrome 56, Mozilla Firefox 51, Opera 43, Safari 12.

5 ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

5.1 Разрабатываемые программные модули должны быть самодокументированы, т.е. тексты программ должны содержать все необходимые комментарии.

5.2 В состав сопровождающей документации должны входить:

5.2.1 Расчетно-пояснительная записка на 25-30 листах формата А4 (без приложений 5.2.2 и 5.2.3).

5.2.2 Техническое задание (Приложение А).

5.2.3 Руководство пользователя (Приложение Б).

5.3 Графическая часть должна быть включена в расчетно-пояснительную записку в качестве иллюстраций:

5.3.1 Диаграмма вариантов использования.

5.3.2 Концептуальная диаграмма классов.

5.3.3 Диаграмма состояний интерфейса.

5.3.4 Схема структурная программного обеспечения.

5.3.5 Схема иерархии меню.

5.3.6 Диаграмма классов уровня реализации.

5.3.7 Диаграмма компоновки.

5.3.8 Схема базы данных.

5.3.9 Таблицы тестов.

6 СТАДИИ И ЭТАПЫ РАЗРАБОТКИ

Этап	Содержание этапа	Сроки и объем	Представляемые результаты	
			Спецификации и программный продукт	Документы
1.	Выбор темы, составление задания, решение организационных вопросов	1..2 недели (10 %)	-	Заполненный бланк задания на курсовую работу – вывешивается на сайт кафедры для получения утверждающей подписи заведующего кафедрой
2.	Анализ предметной области, разработка ТЗ. Исследование методов решения, выбор основных проектных решений	3..4 недели	Результаты декомпозиции предметной области. Эскизный проект: интерфейс, схемы, возможно, часть программы (выбранные готовые решения).	Фрагмент расчетно-пояснительной записки с обоснованием выбора средств и подходов к разработке

Этап	Содержание этапа	Сроки и объем	Представляемые результаты	
			Спецификации и программный продукт	Документы
3.	Сдача ТЗ	4 неделя (25 %)	-	Техническое задание – утверждается руководителем
4.	Проектирование и реализация основных компонентов – ядра программы	5..7 недели	Технический проект основной части: структура программы, алгоритмы программ, описания структур данных, диаграмма классов – в зависимости от выбранной технологии разработки. Программный продукт, реализующий основные функции (демонстрируется руководителю)	Фрагмент расчетно-пояснительной записки с обоснованием разработанных спецификаций Тексты части программного продукта, реализующего основные функции.
5.	Сдача прототипа программного продукта	7 неделя (50 %)	Прототип программного продукта – демонстрируется руководителю	
6.	Разработка компонентов, обеспечивающих функциональную полноту	8..10	Рабочий проект программы. Готовая программа	Черновик расчетно-пояснительной записки. Тексты программного продукта.
7.	Сдача программного продукта	11 неделя (75 %)	Готовая программа – оценивается руководителем в баллах	-
8.	Тестирование программы и подготовка документации	12..14	Тесты и результаты тестирования.	РПЗ и Руководство пользователя.
9.	Оформление и сдача документации	14 неделя (90 %)	–	Расчетно-пояснительная записка и Руководство пользователя – проверяются и подписываются руководителем
10.	Защита курсовой работы	15..16 недели (100%)	–	Доклад (3-5 минут). Защита курсовой работы. Подписанная документация – вывешивается на сайт кафедры

7 ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ

7.1 Порядок контроля

Контроль выполнения осуществляется руководителем еженедельно.

7.2 Порядок защиты

Защита осуществляется комиссии преподавателей кафедры.

7.3 Срок защиты

Срок защиты: 15-16 недели.

8 ПРИМЕЧАНИЕ

В процессе выполнения работы возможно уточнение отдельных требований технического задания по взаимному согласованию руководителя и исполнителя.

ПРИЛОЖЕНИЕ Б

Руководство пользователя

**Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)


НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.01 Информатика и вычислительная техника

ВЕБ-ПРИЛОЖЕНИЕ «СУА» ДЛЯ ОРГАНИЗАЦИИ ОЧЕРЕДИ НА ОБСЛУЖИВАНИЕ

Руководство пользователя

Листов 18

Студент ИУ6-52Б
(Группа)

 09.12.2021
(Подпись, дата)

Р.В. Баканов
(И.О. Фамилия)

Руководитель курсовой работы,
(д.т.н., проф.)

 09.12.2021
(Подпись, дата)

Г.С. Иванова
(И.О. Фамилия)

Москва, 2021

Содержание

1 Общие сведения о программном продукте	48
2 Начало работы с приложением	49
3 Инструкция по работе с приложением	50
3.1 Регистрация и авторизация	50
3.2 Работа с комнатами.....	51
3.3 Работа с очередями	55
3.4 Работа с расписаниями	58
4 Общие рекомендации	61

1 Общие сведения о программном продукте

Программный продукт предоставляет пользователю возможность создать расписание или организовать и контролировать продвижение онлайн-очереди. Веб-приложение позволяет создавать комнаты и присоединяться к уже существующим. Внутри комнат пользователь может создавать непосредственно очереди и расписания. Предусмотрена возможность редактирования информации о своей учетной записи, созданных комнатах, очередях и расписаниях.

В приложении реализована система авторизации, позволяющая создавать новых пользователей, входить в систему уже существующим пользователем и восстанавливать пароль при необходимости.

Работа с веб-приложением «Cua» может осуществляться с помощью персонального компьютера, ноутбука или смартфона, имеющих доступ к сети Интернет.

2 Начало работы с приложением

Для начала работы с веб-приложением необходимо иметь стабильное Интернет-соединение, установленный браузер и перейти по адресу одной из веб-страниц, реализованных в веб-приложении. После запуска система перенаправит неавторизованного пользователя на страницу авторизации, где он сможет ввести данные своей учетной записи и получить доступ к основным возможностям веб-приложения. Также для получения этого доступа неавторизованный пользователь может создать новую учетную запись или восстановить пароль в случае его потери.

3 Инструкция по работе с приложением

3.1 Регистрация и авторизация

Страница авторизации представляет из себя форму с двумя полями – полем для электронной почты и полем для пароля. Пользователь заполняет эти поля и нажимает кнопку «Войти». В случае корректно введенных данных он перенаправляется на главную страницу веб-приложения.

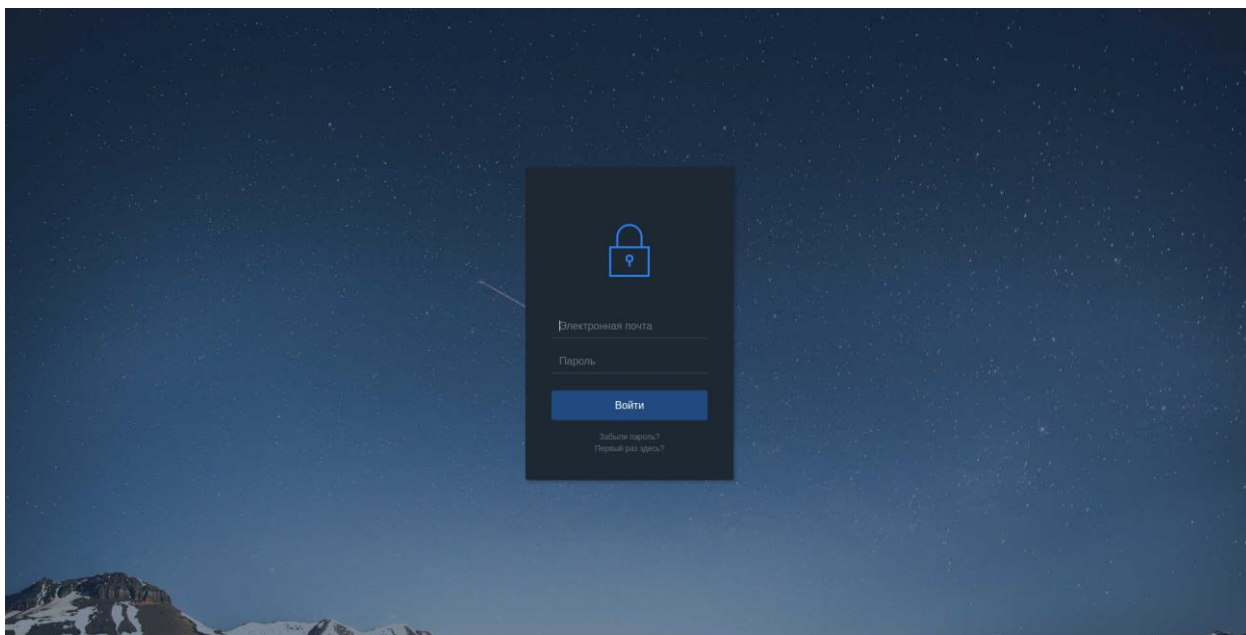


Рисунок Б.1 – Страница авторизации

Если у пользователя еще нет учетной записи, он может создать новую, перейдя по ссылке «Первый раз здесь?». Страница регистрации также представляет из себя форму, но с расширенным списком полей.

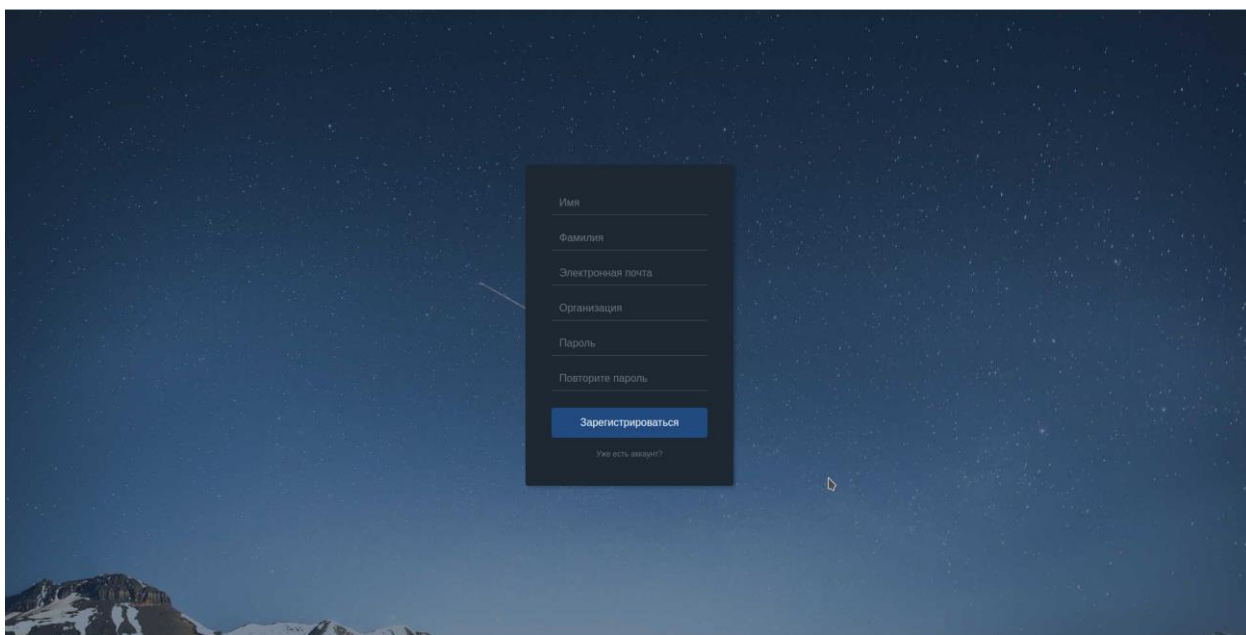


Рисунок Б.2 – Страница регистрации

В случае корректно введенных регистрационных данных и нажатия кнопки «Зарегистрироваться» пользователю на указанный адрес электронной почты приходит письмо, содержащее ссылку для подтверждения новой учетной записи.

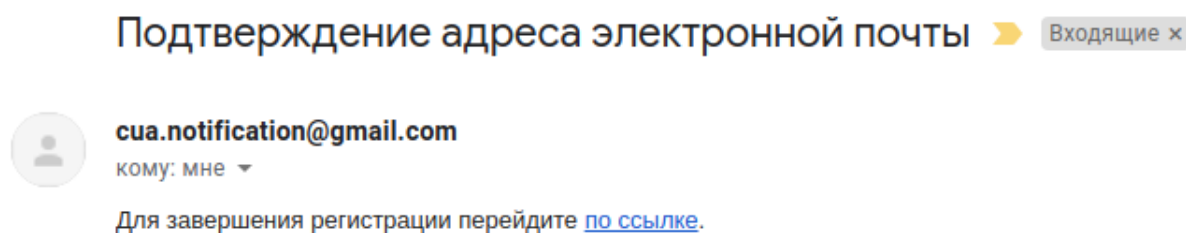


Рисунок Б.3 – Письмо с подтверждением

3.2 Работа с комнатами

На главной странице отображаются те комнаты, к которым пользователь имеет какое-либо отношение (созданные им комнаты и комнаты, в которые он вступил). Карточка комнаты содержит следующую информацию:

- название комнаты;
- имя администратора;
- название организации;
- количество участников;
- информация о закрытости / открытости комнаты;
- информация о видимости / невидимости комнаты;
- список очередей и расписаний.

Список комнат можно отсортировать по имени или по тому, были ли они созданы текущим пользователем или нет. Для этого необходимо нажать кнопку «Сортировать». Возможно также отфильтровать комнаты по имени, для этого достаточно начать вводить название комнаты в поле «Найти комнату...».

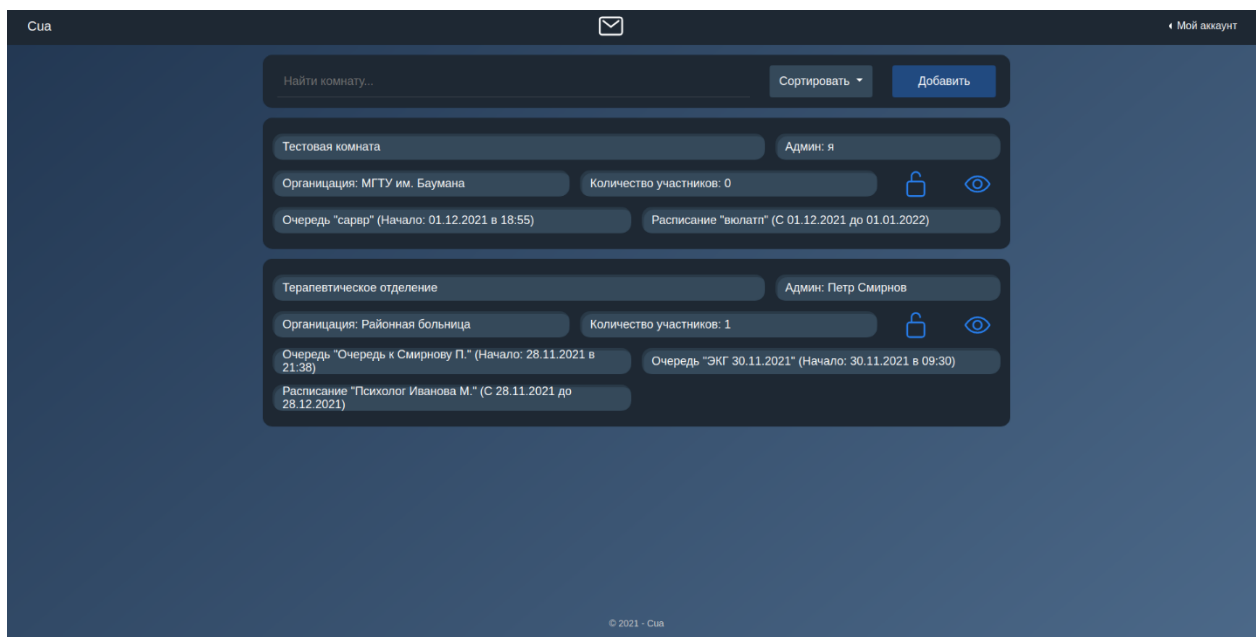


Рисунок Б.4 – Главная страница

Чтобы создать новую комнату нужно нажать кнопку «Добавить», а потом выбрать «Создать». Это перенаправит пользователя на страницу создания комнаты, где ему нужно будет заполнить поля формы и нажать кнопку «Создать», после чего он будет перенаправлен обратно на главную страницу.

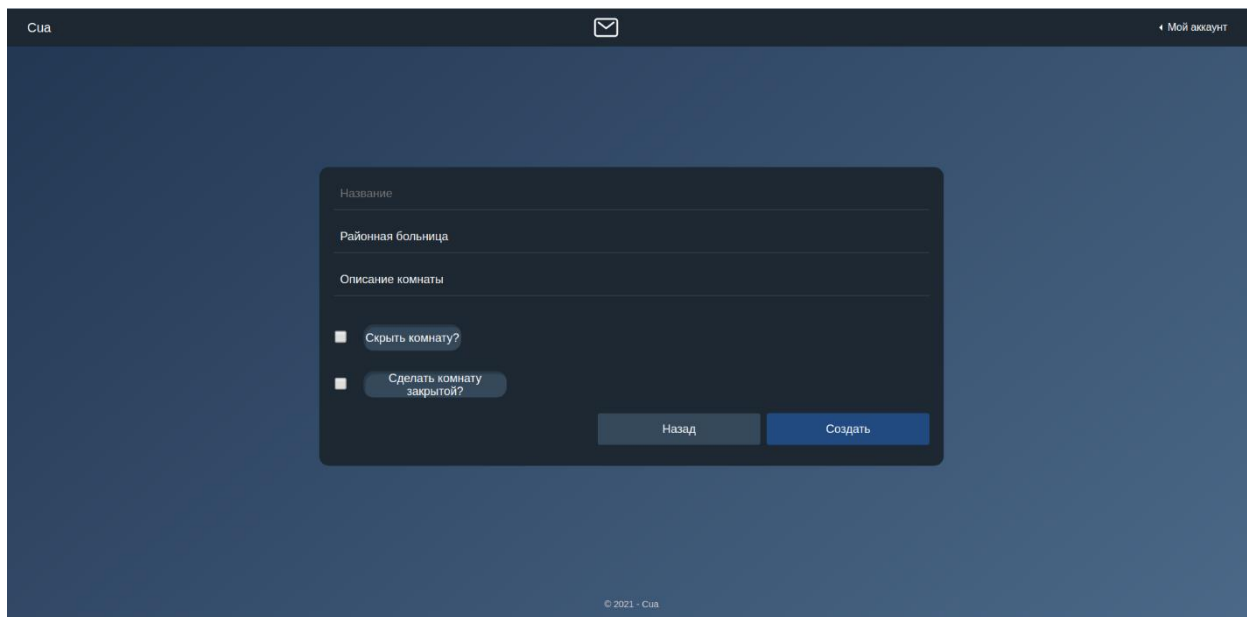


Рисунок Б.5 – Страница создания комнаты

Чтобы присоединиться к уже существующей комнате нужно нажать кнопку «Добавить», а потом выбрать «Вступить». Это перенаправит пользователя на страницу доступных комнат, где он сможет присоединиться к какой-либо комнате по нажатию кнопки «Присоединиться». Список доступных комнат можно отфильтровать по названию, имени администратора, названию организации или по тому, является ли комната закрытой

или нет. Чтобы вернуться обратно на главную страницу нужно щелкнуть по иконке «Cua» в левой верхней части экрана.

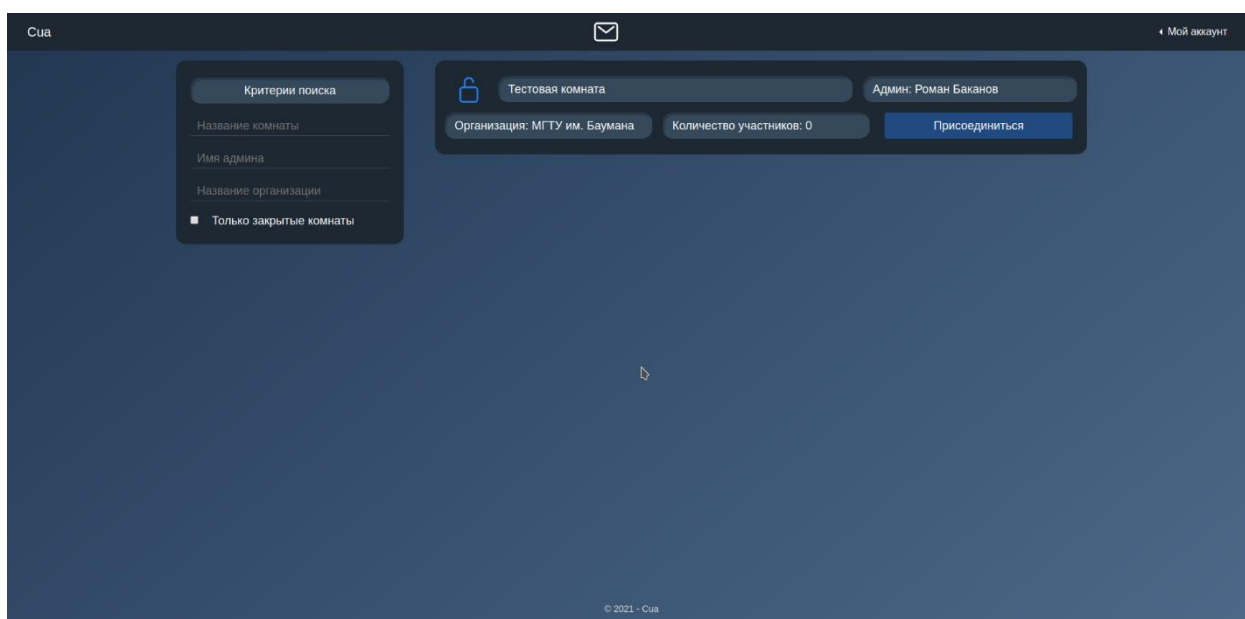


Рисунок Б.6 – Страница доступных комнат

Нажав на карточку комнаты на главной странице, пользователь попадает на страницу комнаты. Здесь пользователь может писать сообщения в чат комнаты, заполнив поле «Ваше сообщение» и нажав кнопку «Отправить». Также есть возможность ознакомиться со списком очередей и расписаний. Для этого нужно нажать кнопку «Очереди и расписания».

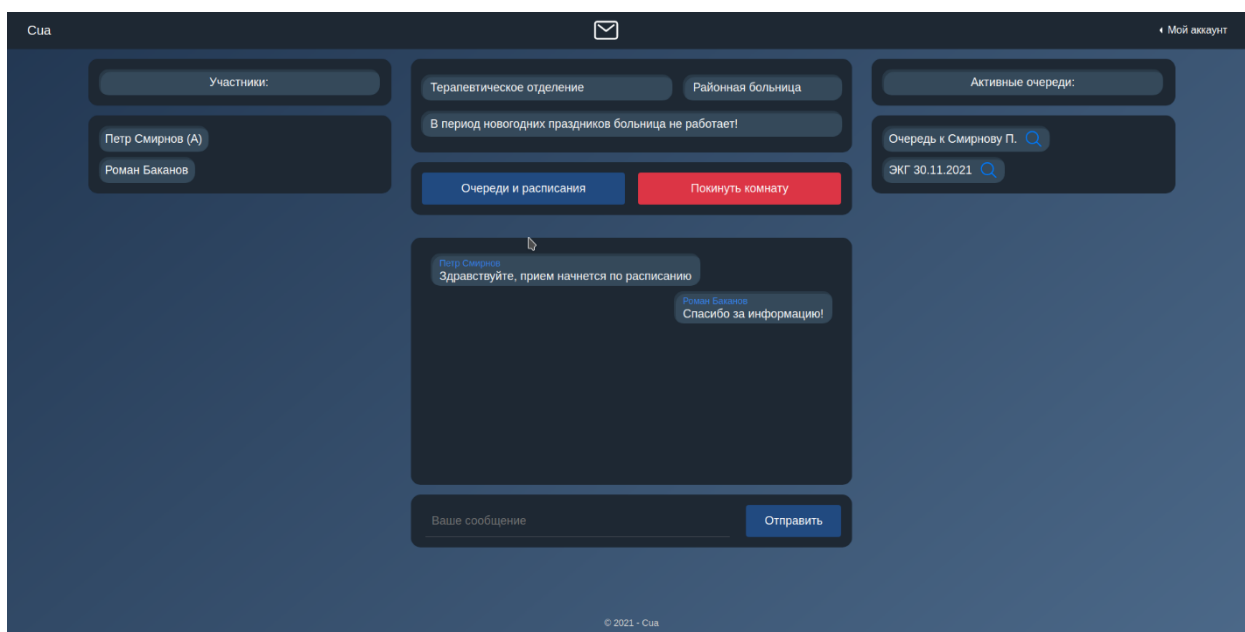


Рисунок Б.7 – Страница комнаты (вид для обычного пользователя)

Для администраторов и модераторов на этой странице присутствует дополнительная кнопка «Панель управления», невидимая обычным пользователям. Нажав на нее администратор или модератор попадает на страницу управления комнатой.

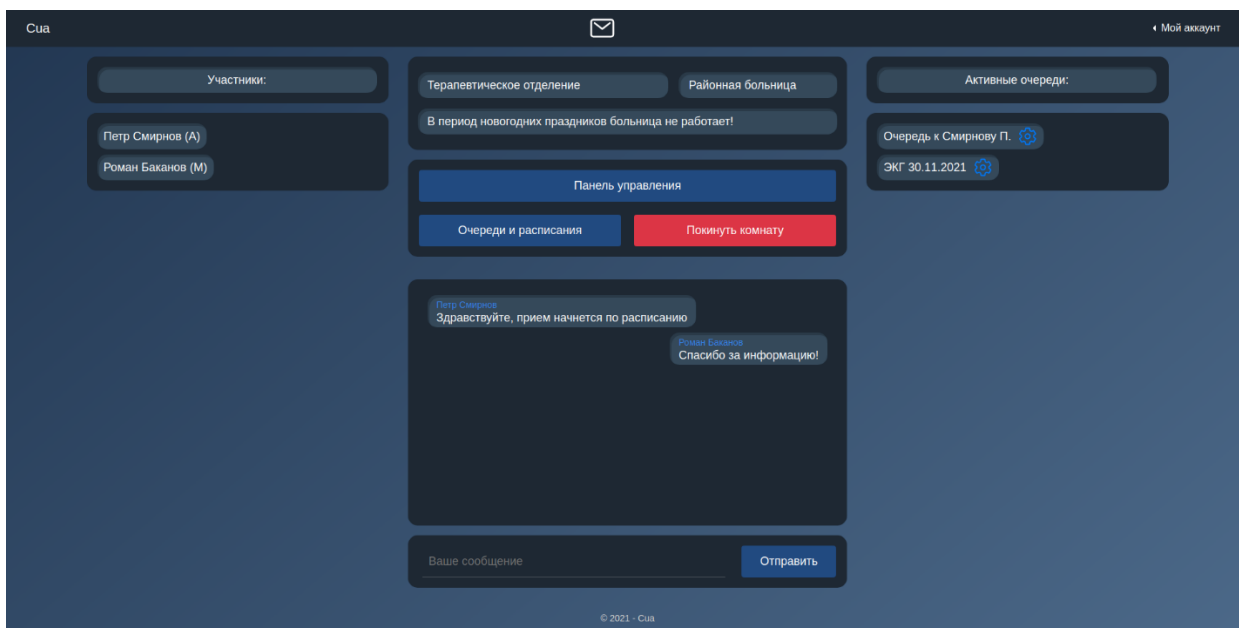


Рисунок Б.8 – Страница комнаты (вид для модератора)

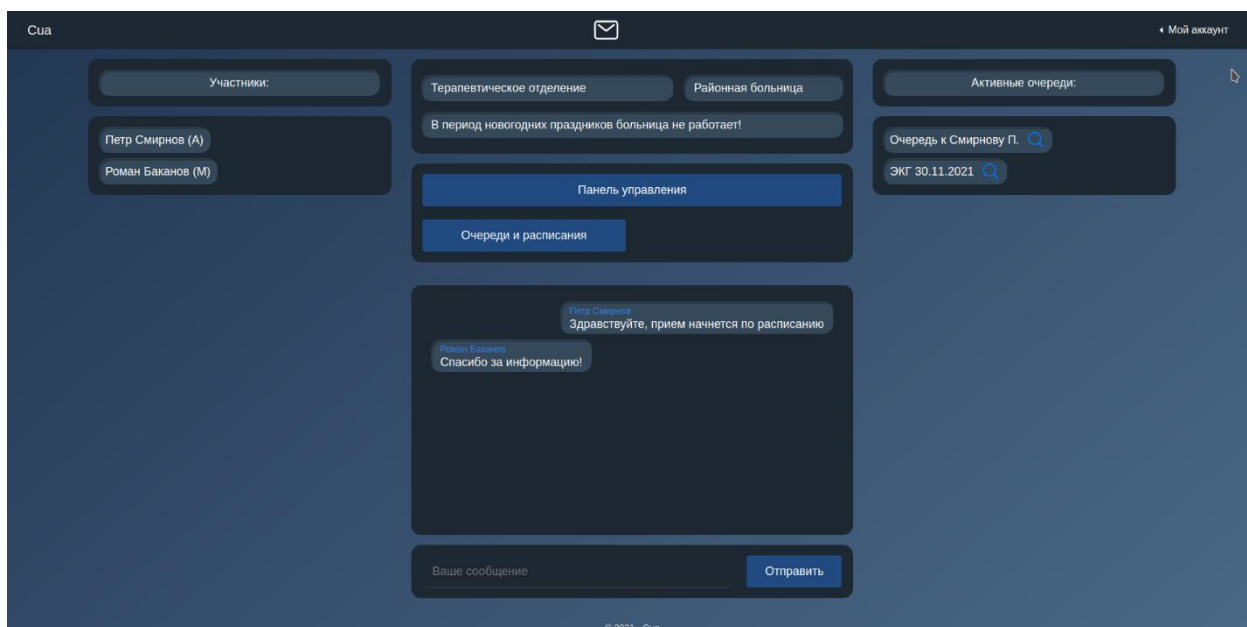


Рисунок Б.9 – Страница комнаты (вид для администратора)

На странице управления комнатой администратор может повысить пользователя до модератора, нажав на иконку стрелки вверх, или, наоборот, понизить до обычного пользователя, нажав на иконку стрелки вниз. Также он может удалить пользователя из комнаты, нажав на иконку крестика. Есть возможность изменить описание комнаты путем нажатия кнопки «Изменить описание». Это заменит поля с информацией на поля формы, заполнив которые пользователь может сохранить изменения, нажав кнопку «Сохранить». Нажатие кнопки «Удалить комнату» приводит к безвозвратному удалению комнаты. Перед этим действием у пользователя запрашивается подтверждение.

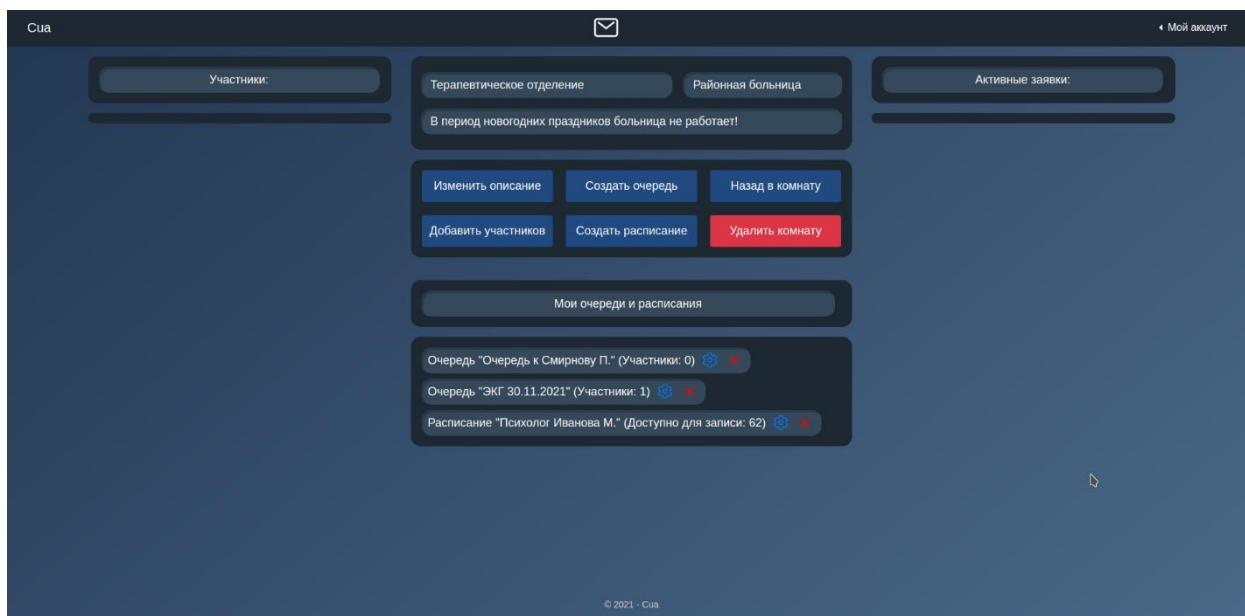


Рисунок Б.10 – Страница управления комнатой

Кнопка «Добавить участников» перенаправляет пользователя на страницу рассылки приглашений. На этой странице администратор может найти нужного пользователя по имени или названию организации и пригласить его в комнату, нажав кнопку «Пригласить».

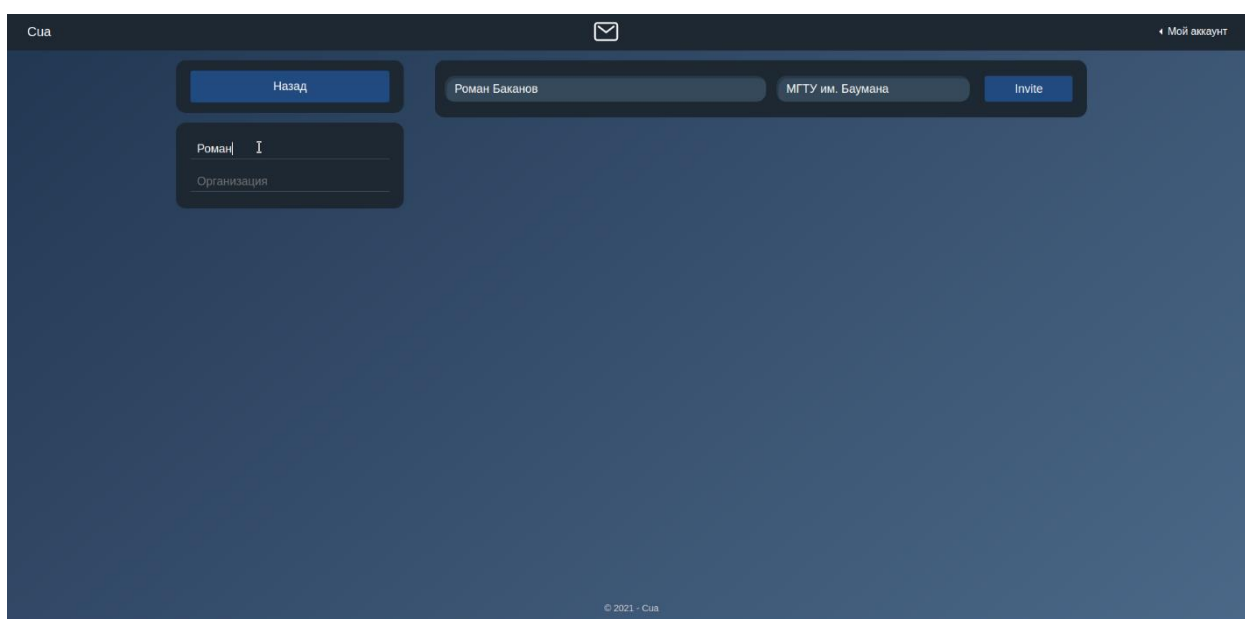


Рисунок Б.11 – Страница рассылки приглашений

3.3 Работа с очередями

На странице управления комнатой есть кнопка «Создать очередь». Нажав ее, пользователь попадает на страницу создания очереди, которая представляет из себя форму со следующими полями:

- название;
- ограничение на количество участников;

— время начала.

Рисунок Б.12 – Страница создания очереди

Заполнив поля и нажав кнопку «Создать», пользователь возвращается на страницу управления комнатой, где только что созданная очередь появляется в списке «Мои очереди и расписания». Нажав на иконку шестеренки напротив названия очереди, пользователь попадает на страницу настройки очереди.

Рисунок Б.13 – Страница настройки очереди

На этой странице он может удалить участников очереди и изменить основную информацию. Если же нажать иконку крестика напротив названия очереди, то она удалится.

На странице очередей и расписаний пользователь может ознакомиться со всеми очередями и расписаниями, созданными в этой комнате. Этот список можно отфильтровать

по названию и по типу (очередь или расписание). На карточки очереди присутствует основная информация:

- тип;
- название;
- количество участников;
- ограничение на количество участников;
- время начала;
- имя создателя.

Создатель комнаты может перевести очередь в активное состояние (то есть начать прием по очереди), нажав кнопку «Начать», или, наоборот, остановить прием, нажав кнопку «Остановить». Любой пользователь может ознакомиться со списком участников по нажатию кнопки «Показать участников» или присоединиться к очереди по нажатию кнопки «Присоединиться». Чтобы покинуть очередь достаточно нажать кнопку «Покинуть».

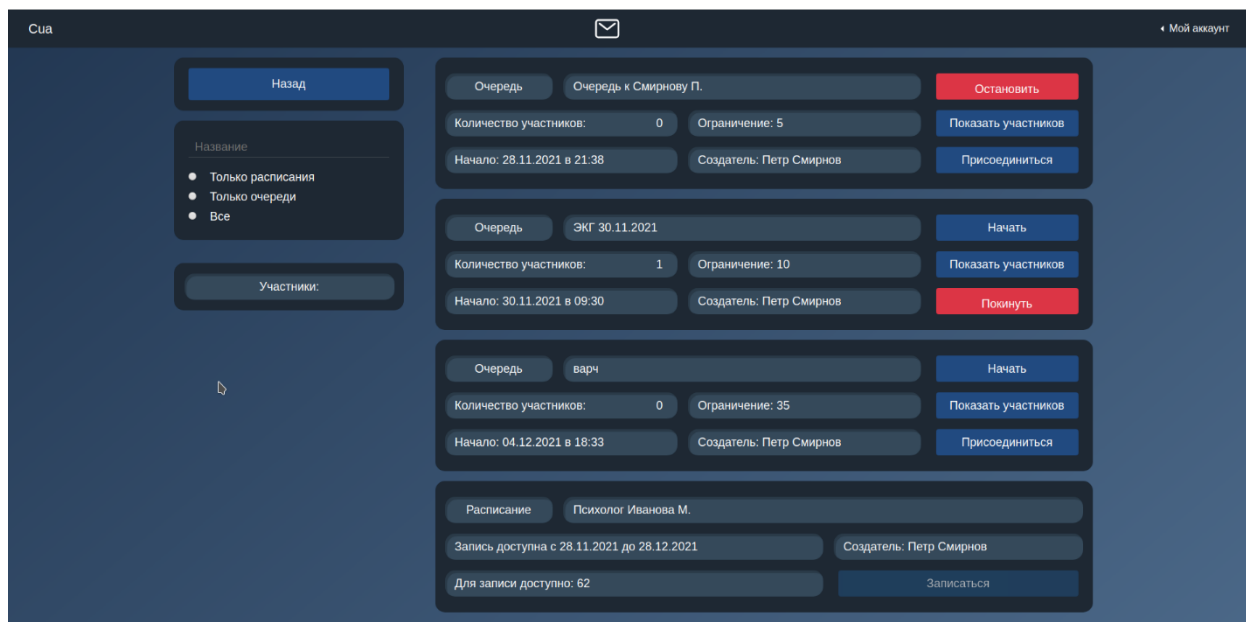


Рисунок Б.14 – Очереди на странице очередей и расписаний

На странице комнаты присутствует список активных очередей, с помощью которого создатель очереди может перейти на панель управления очередью, а обычный пользователь – посмотреть состояние очереди. Для этого необходимо нажать на иконку шестеренки или лупы соответственно. Создатель комнаты в рамках управления активной очередью может контролировать процесс продвижения очередью с помощью кнопки «Пригласить следующего».

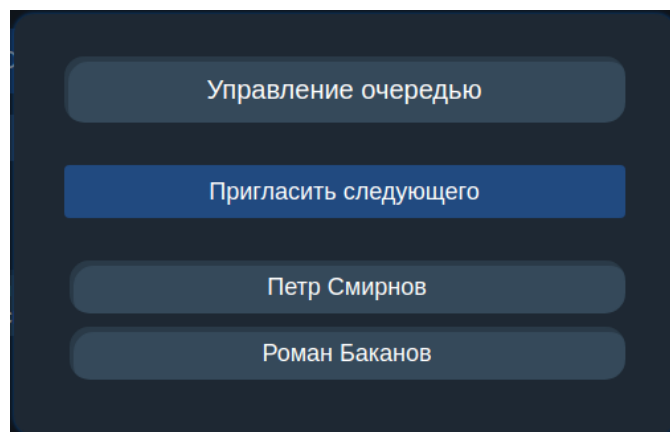


Рисунок Б.15 – Панель управления очередью

3.4 Работа с расписаниями

Чтобы создать расписание, надо нажать кнопку «Создать расписание» на странице управления комнатой. Далее необходимо заполнить поля формы и нажать кнопку «Создать».

Рисунок Б.16 – Страница создания расписания

Последним шагом в процессе создания расписания является назначение приемов. Для этого на странице назначения приемов необходимо выбрать промежутки времени, в которые будет вестись прием, и нажать кнопку со стрелкой. Это вернет нас на страницу управления комнатой, а новая очередь будет помещена в список «Мои очереди и расписания».

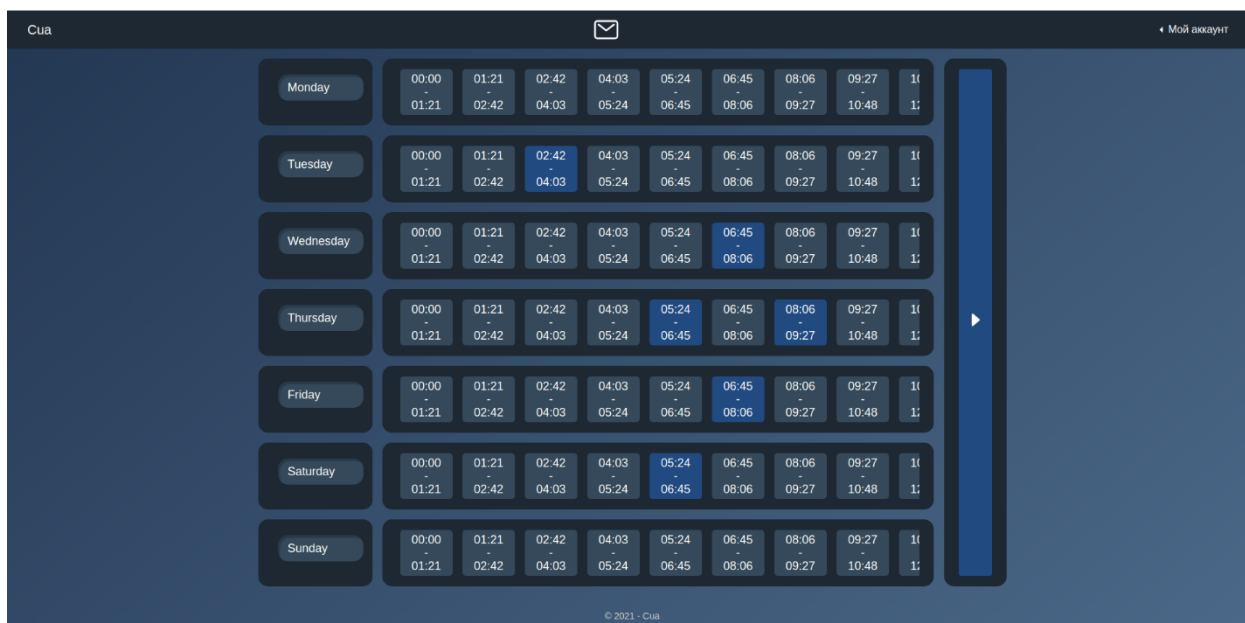


Рисунок Б.17 – Страница назначение приемов

Нажав на иконку шестеренки напротив названия расписания, пользователь попадает на страницу настройки расписания, где он может отменить записи по нажатию иконки крестика напротив соответствующей записи. Также пользователь может изменить название расписания и продлить запись. Для этого используются кнопки «Изменить название» и «Продлить запись» соответственно.

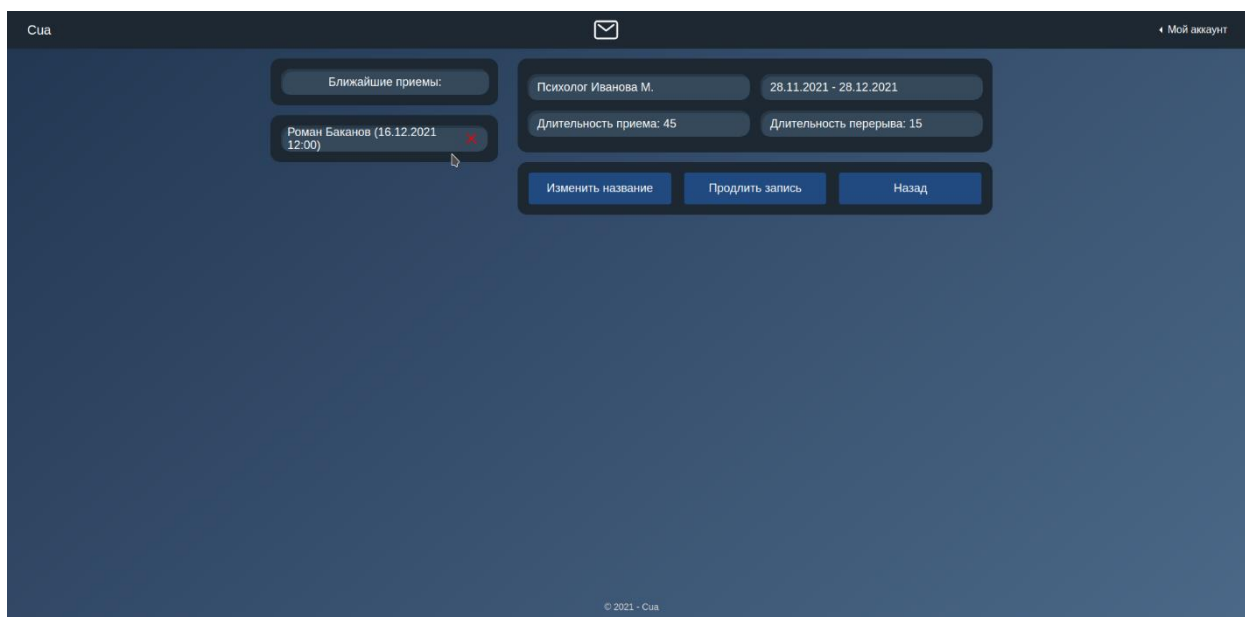
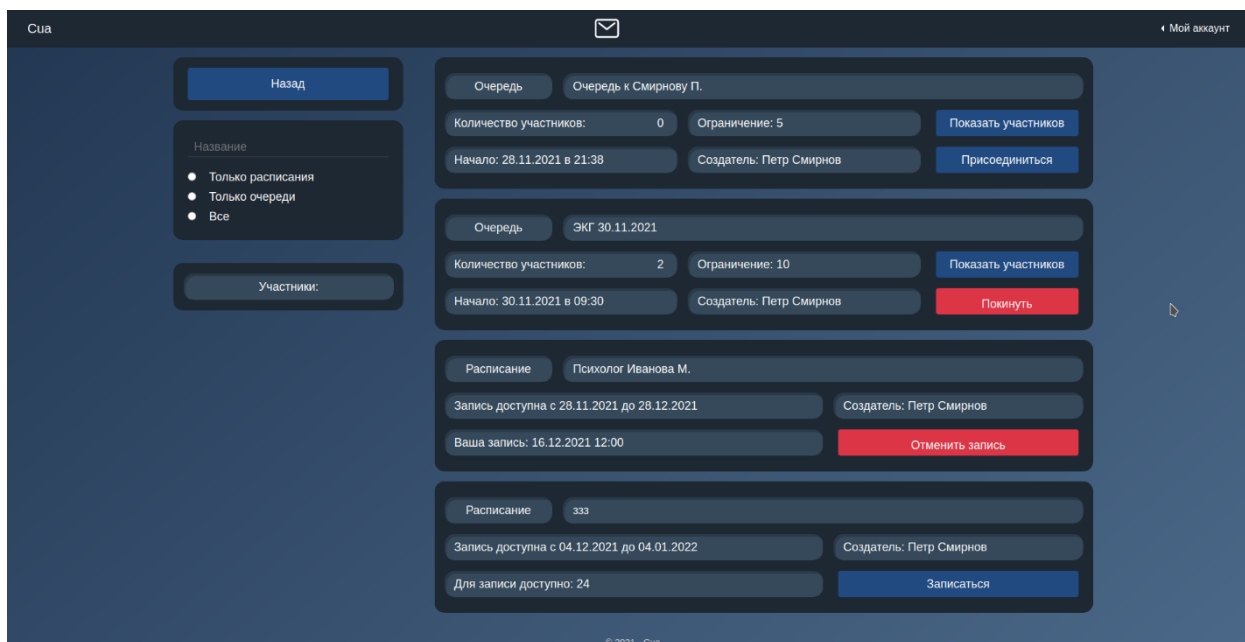


Рисунок Б.18 – Страница настройки расписания

Карточка расписания на странице со списком очередей и расписаний содержит следующую информацию:

- тип;
- название;

- даты, доступные для записи;
- имя создателя;
- количество приемов, доступных для записи.



Б.19 – Расписания на странице очередей и расписаний

Пользователь может записаться на прием, нажав на кнопку «Записаться». Нажатие этой кнопки приведет к появлению на экране формы с двумя полями – датой приема и временем приема. Заполнив эти поля, пользователь должен нажать кнопку «Подтвердить».

Рисунок Б.20 – Запись на прием

Отменить запись можно нажатием кнопки «Отменить запись».

4 Общие рекомендации

В заключении следует рассмотреть общие рекомендации по использованию веб-приложения «Cua».

Веб-приложение не будет функционировать без подключения к сети Интернет. В этом случае при попытке входа или работы с данными будут получены ошибки. При возникновении подобной проблемы рекомендуется проверить сетевое подключение на своем устройстве.

При регистрации необходимо указывать реально существующий адрес электронной почты, так как на нее будет высылаться письмо с подтверждением учетной записи.