Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Волгоградский государственный технический университет»

Факультет \_\_\_Экономики и управления\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра \_\_Информационные системы в экономике\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовой работе**

по дисциплине Проектирование информационных систем \_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

на тему\_\_Разработка приложения для салона красоты\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент\_\_ Чударь Альбина Олеговна \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, имя, отчество)

Группа\_\_ЭП-462\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель работы (проекта) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_Н.В. Кетько\_\_\_\_\_

(подпись и дата подписания) (инициалы и фамилия)

Члены комиссии:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись и дата подписания) (инициалы и фамилия)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись и дата подписания) (инициалы и фамилия)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись и дата подписания) (инициалы и фамилия)

Нормоконтролер \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата подписания) (инициалы и фамилия)

Волгоград 2025 г.

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Волгоградский государственный технический университет»

Факультет \_\_\_Экономики и управления\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Направление (специальность)\_\_\_Прикладная информатика в экономике \_\_\_\_  
Кафедра \_\_Информационные системы в экономике\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дисциплина\_\_ Проектирование информационных систем\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
|  | Утверждаю  Зав. кафедрой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | «\_\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2025 г. |

**Задание**

**на курсовую работу**

Студент\_\_ Чударь Альбина Олеговна\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, имя, отчество)

Группа\_\_ЭП-462\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Тема: \_\_ Разработка приложения для салона красоты\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Утверждена приказом от «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2025\_ г. № \_\_\_\_\_\_\_\_\_

2. Срок представления работы (проекта) к защите «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2025 г.

3. Содержание расчетно-пояснительной записки: Введение; Описание модели приложения; Описание программного решения и примеры кода; Экранные\_\_ формы приложения; Заключение.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Перечень графического материала: \_14 рисунков\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Дата выдачи задания «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20 25\_ г.

Руководитель работы (проекта)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_Н.В. Кетько\_\_\_\_\_

подпись, дата инициалы и фамилия

Задание принял к исполнению\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_А.О. Чударь\_\_\_

подпись, дата инициалы и фамилия

Содержание

[ВВЕДЕНИЕ 4](#_Toc189351153)

[ГЛАВА I. ОПИСАНИЕ МОДЕЛИ ПРИЛОЖЕНИЯ 6](#_Toc189351154)

[ГЛАВА II. ПОСТАНОВКА ЗАДАЧИ 11](#_Toc189351155)

[ГЛАВА III. ОПИСАНИЕ КОДА И ЭКРАННЫЕ ФОРМЫ 12](#_Toc189351156)

[3.1. Описание программного решения и элементы кода 12](#_Toc189351157)

[3.2. Экранные формы 22](#_Toc189351158)

[ЗАКЛЮЧЕНИЕ 26](#_Toc189351159)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 27](#_Toc189351160)

ВВЕДЕНИЕ

Java Swing — это мощный и универсальный набор библиотек и компонентов, предназначенных для создания графических пользовательских интерфейсов (GUI) в приложениях на языке Java. Впервые представленная в версии Java 1.2, эта технология стала важным инструментом для разработчиков, позволяя создавать современные, интерактивные и визуально привлекательные приложения. Благодаря своей гибкости и кроссплатформенности, Java Swing остается актуальным решением для разработки ПО даже в условиях появления новых технологий.

Современные технологии в сфере разработки программного обеспечения играют ключевую роль в улучшении и упрощении различных аспектов жизни. Одной из областей, где программные решения становятся все более востребованными, является индустрия wellness и спа-услуг. Развитие технологий позволяет создавать удобные и функциональные приложения, которые помогают спа-салонам оптимизировать свою работу, улучшить взаимодействие с клиентами и предоставлять более качественный сервис, направленный на релаксацию и восстановление.

Целью данной курсовой работы является разработка приложения для управления услугами спа-салона с использованием технологии Java Swing. Приложение ориентировано на клиентов спа-салона. Оно предоставляет удобные инструменты для записи на процедуры, управления расписанием, учета клиентов. Это позволит упростить процесс организации работы спа-салона, повысить уровень сервиса и создать комфортные условия для клиентов, стремящихся к релаксации и уходу за собой.

Актуальность темы обусловлена растущими требованиями к индустрии wellness и возможностями, которые предоставляют современные технологии. С увеличением популярности спа-услуг и стремлением людей к здоровому образу жизни, спа-салоны стремятся внедрять инновационные решения для улучшения качества обслуживания и привлечения новых клиентов. Приложение на Java Swing может стать эффективным инструментом, предлагая интуитивно понятный интерфейс для записи на процедуры, управления расписанием и учета клиентской базы. Кроме того, использование Java Swing обеспечивает кроссплатформенность, что делает приложение доступным для пользователей различных операционных систем.

Основные задачи курсовой работы включают:

1. Описание модели приложения и формулировку постановки задачи;
2. Разработку программных решений и представление ключевых фрагментов кода;
3. Демонстрацию экранных форм приложения в различных сценариях использования, таких как запись на процедуры, управление расписанием, учет клиентов и анализ загруженности спа-салона.

В рамках работы будет рассмотрен процесс проектирования и реализации графического интерфейса, а также функциональности, необходимой для эффективного управления услугами спа-салона. Приложение позволит клиентам удобно и быстро записываться на процедуры, выбирать подходящие услуги и планировать свои визиты. Это в целом повысит уровень сервиса, создаст атмосферу комфорта и повысит конкурентоспособность спа-салона на рынке wellness-услуг.

ГЛАВА I. ОПИСАНИЕ МОДЕЛИ ПРИЛОЖЕНИЯ

Разработка моделей является неотъемлемой частью создания сложных систем, даже если система относительно небольшая. Важность этого процесса особенно возрастает, если система должна быть точной, производительной, прибыльной или отвечать важным требованиям. В случае с приложением для СПА-салона "Релакс", моделирование помогает визуализировать структуру, поведение и взаимодействие компонентов системы, что упрощает процесс разработки и тестирования.

Методология проектирования систем получила широкое распространение с появлением языка объектного моделирования нового поколения — унифицированного языка моделирования Unified Modeling Language (UML). Этот язык предназначен для визуального моделирования и проектирования информационных систем. UML представляет собой набор правил, по которым нужно рисовать схемы. Зная его, можно быстро создавать универсальные графические представления сложных процессов и структур. Именно поэтому IT-специалисты во время разработки ПО часто используют UML-моделирование и проектирование процессов.

Для приложения были разработаны пять ключевых диаграмм, которые помогают понять структуру, поведение и взаимодействие компонентов системы.

Диаграмма вариантов использования (Use Case Diagram) описывает, какой функционал разрабатываемой программной системы доступен каждой группе пользователей. В нашем случае актером является Клиент, который взаимодействует с системой через такие варианты использования, как выбор филиала, выбор массажа, запись на услугу, просмотр записей и удаление записи. Эта диаграмма помогает понять, как клиент взаимодействует с системой и какие действия он может выполнять.

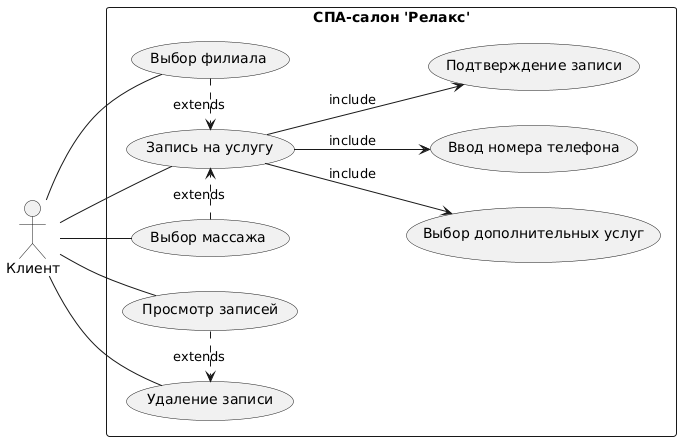


Рисунок 1 - Диаграмма вариантов использования

Диаграмма классов (Class Diagram) описывает структуру системы, показывая классы, их атрибуты, методы и отношения между ними. В нашем приложении выделены два основных пакета: GUI Components, содержащий классы для компонентов интерфейса, и Main Application, содержащий основной класс `SpaBookingApp`, который управляет логикой приложения и взаимодействует с компонентами интерфейса. Эта диаграмма помогает разработчикам понять, как организованы классы и как они взаимодействуют друг с другом.

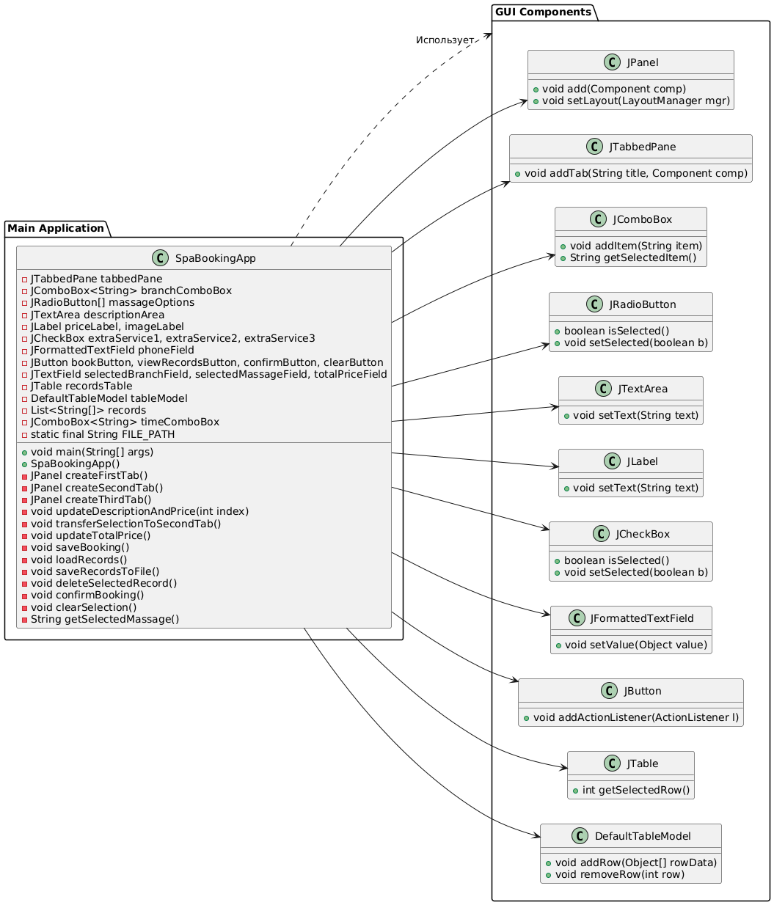


Рисунок 2 - Диаграмма классов

Диаграмма состояний (State Diagram) описывает жизненный цикл объекта или системы, показывая, как объект переходит из одного состояния в другое в ответ на события. В нашем приложении состояния включают выбор филиала, выбор массажа, запись на услугу, подтверждение записи, просмотр записей и удаление записи. Эта диаграмма помогает понять, как система меняет свои состояния в зависимости от действий пользователя.

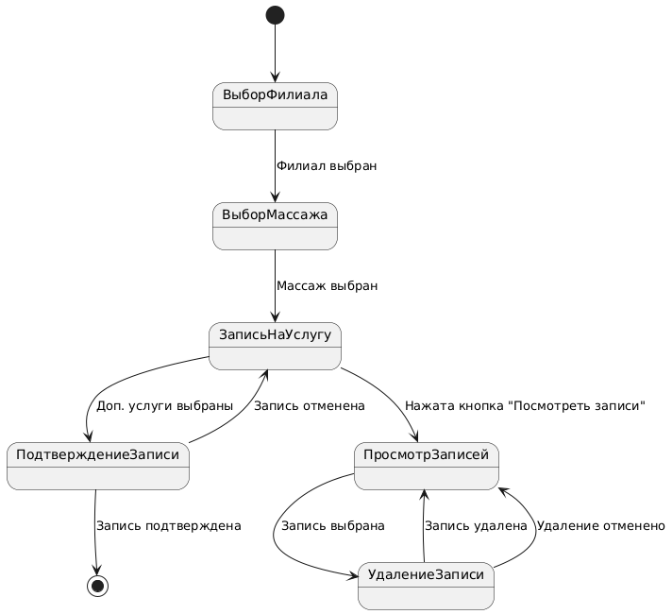


Рисунок 3 - Диаграмма состояний

Диаграмма деятельности с дорожками (Activity Diagram with Swimlanes) описывает поток действий в системе, разделяя их на "дорожки", которые представляют различные роли или компоненты. В нашем приложении дорожки разделены на клиента и систему. Клиент выполняет действия, такие как выбор филиала и массажа, а система отображает данные, обновляет цены и сохраняет записи. Эта диаграмма помогает визуализировать последовательность действий и то, как они распределены между клиентом и системой.

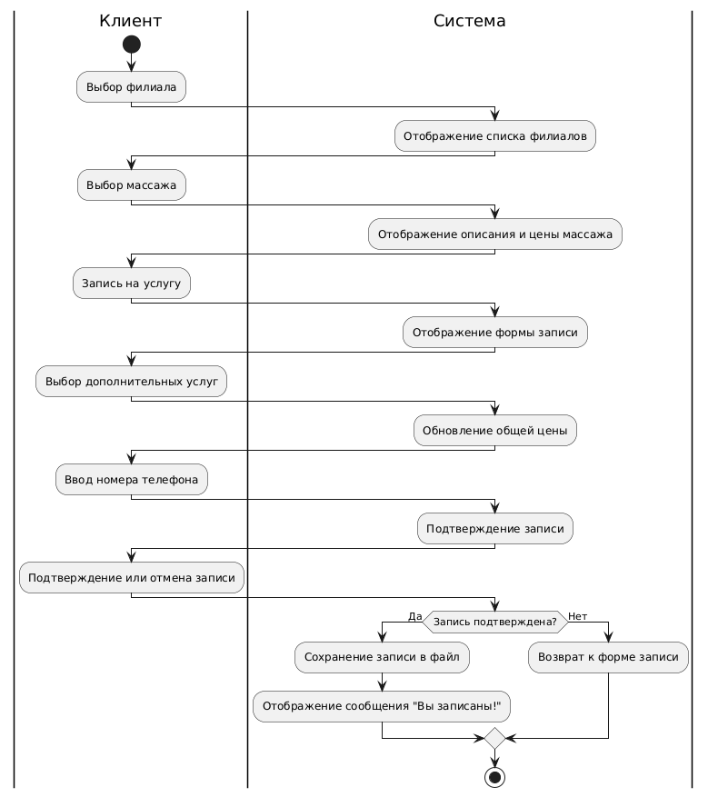


Рисунок 4 - Диаграмма деятельности с дорожками

Диаграмма последовательности (Sequence Diagram) описывает взаимодействие между объектами в системе в виде последовательности сообщений. В нашем приложении она показывает, как клиент выбирает филиал и массаж, система отображает описание и цену, клиент выбирает дополнительные услуги и вводит номер телефона, а система сохраняет запись и отображает подтверждение. Эта диаграмма помогает понять, как объекты взаимодействуют друг с другом в процессе выполнения сценария.

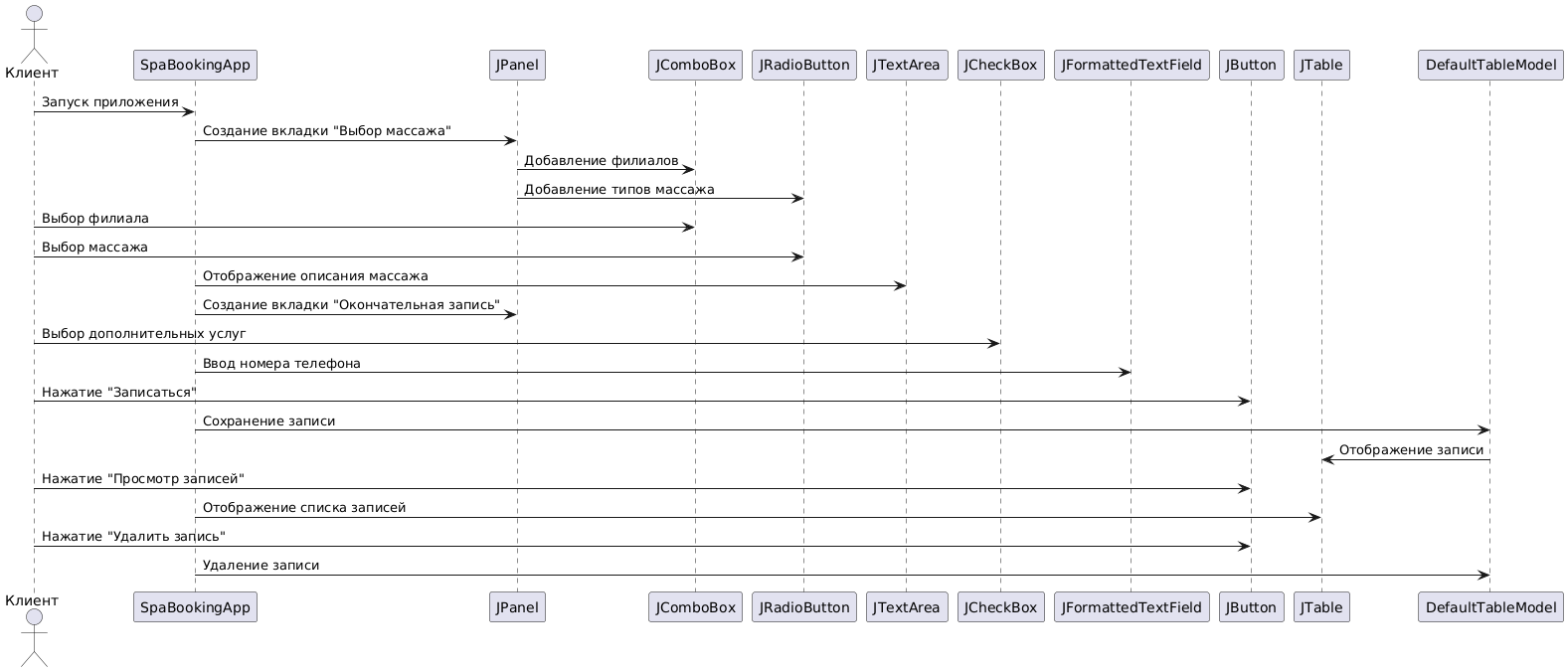


Рисунок 5 - Диаграмма последовательности

Разработанные диаграммы играют ключевую роль в понимании структуры и поведения системы СПА-салона "Релакс". Они помогают визуализировать взаимодействие клиента с системой, определить ключевые компоненты системы и их отношения, описать последовательность действий и состояний системы, а также упростить процесс разработки и тестирования. Использование UML-моделирования позволяет создать четкую и понятную документацию, которая облегчает коммуникацию между разработчиками, дизайнерами и заказчиками. Это особенно важно для создания качественного и удобного приложения, которое будет отвечать всем требованиям клиентов СПА-салона "Релакс".

ГЛАВА II. ПОСТАНОВКА ЗАДАЧИ

Пользователь заходит в приложение для записи на услуги СПА-салона "Релакс". Приложение предоставляет возможность выбрать один из нескольких филиалов салона. После выбора филиала пользователю становится доступен список доступных видов массажа. Каждый вид массажа сопровождается подробным описанием. Пользователь может ознакомиться с описанием и выбрать подходящий для себя вариант.

После выбора массажа пользователь переходит на вкладку "Окончательная запись". На этой вкладке пользователю предлагается выбрать дополнительные услуги, каждая из которых имеет свою стоимость. Пользователь может включить или исключить эти услуги, что сразу отразится на общей цене. Кроме того, пользователь может ввести свой номер телефона для связи и выбрать удобное время для записи.

Приложение автоматически рассчитывает общую стоимость выбранных услуг. Если пользователя всё устраивает, он может подтвердить запись, после чего система проверяет введённые данные и уведомляет пользователя об успешной записи. Если пользователь хочет изменить свои предпочтения или записаться на другую услугу, он может очистить текущий выбор и повторить процесс заново.

Приложение также предоставляет возможность просмотра всех текущих записей пользователя. Пользователь может удалить любую запись, если она больше не актуальна. Удаление происходит после подтверждения, чтобы избежать случайных действий.

Основная цель приложения — сделать процесс записи на услуги СПА-салона максимально удобным и прозрачным. Пользователь может легко выбрать подходящие услуги, ознакомиться с их описанием и стоимостью, а также управлять своими записями. Приложение также учитывает возможность изменения выбора. Всё это делает процесс взаимодействия с салоном комфортным и приятным, что способствует повышению удовлетворённости клиентов и их лояльности к салону "Релакс".

ГЛАВА III. ОПИСАНИЕ КОДА И ЭКРАННЫЕ ФОРМЫ

* 1. Описание программного решения и элементы кода

Для разработки приложения используется встроенная в язык программирования Java библиотека Swing, которая предоставляет компоненты для создания графического пользовательского интерфейса (GUI). Приложение представляет собой систему для записи на массаж в СПА-салоне, где пользователь может выбрать филиал, тип массажа, дополнительные услуги, указать контактные данные и время записи. Приложение также позволяет просматривать и удалять существующие записи.

Основные компоненты приложения:

1. Главное окно (JFrame):

Главное окно приложения создается с помощью класса Frame. Оно является контейнером для всех остальных компонентов. Внутри окна используется JTabbedPane для организации интерфейса в виде вкладок.

public SpaBookingApp**()** **{**

setTitle**(**"СПА-салон 'Релакс'"**);**

setSize**(**1000**,** 700**);**

setDefaultCloseOperation**(**JFrame**.**EXIT\_ON\_CLOSE**);**

setLayout**(new** BorderLayout**());**

// Инициализация tableModel

String**[]** columnNames **=** **{**"Филиал"**,** "Массаж"**,** "Доп. услуги"**,** "Цена"**,** "Телефон"**,** "Действие"**};**

tableModel **=** **new** DefaultTableModel**(**columnNames**,** 0**);**

tabbedPane **=** **new** JTabbedPane**();**

records **=** **new** ArrayList**<>();**

loadRecords**();**

tabbedPane**.**addTab**(**"Выбор массажа"**,** createFirstTab**());**

tabbedPane**.**addTab**(**"Окончательная запись"**,** createSecondTab**());**

tabbedPane**.**addTab**(**"Мои записи"**,** createThirdTab**());**

add**(**tabbedPane**,** BorderLayout**.**CENTER**);**

setVisible**(true);**

**}**

- setTitle("СПА-салон 'Релакс'"): Устанавливает заголовок окна.

- setSize(1000, 700): Задает размер окна.

- setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE): Закрывает приложение при нажатии на крестик.

- tabbedPane.addTab(...): Добавляет три вкладки в приложение: "Выбор массажа", "Окончательная запись" и "Мои записи".

1. Первая вкладка (Выбор массажа):

На этой вкладке пользователь выбирает филиал и тип массажа. Вкладка состоит из нескольких панелей и компонентов:

private JPanel createFirstTab**()** **{**

JPanel panel **=** **new** JPanel**(new** BorderLayout**());**

panel**.**setBackground**(**Color**.**decode**(**"#FFEFD5"**));**

// Постоянное изображение сверху

JLabel topImageLabel **=** **new** JLabel**(new** ImageIcon**(**"D:/вуз/4 курс/Кетько/курсовая/0.png"**));**

topImageLabel**.**setHorizontalAlignment**(**JLabel**.**CENTER**);**

Image imgTop **=** **((**ImageIcon**)** topImageLabel**.**getIcon**()).**getImage**().**getScaledInstance**(**300**,** 300**,** Image**.**SCALE\_SMOOTH**);**

topImageLabel**.**setIcon**(new** ImageIcon**(**imgTop**));**

panel**.**add**(**topImageLabel**,** BorderLayout**.**NORTH**);**

// Заголовок

JLabel titleLabel **=** **new** JLabel**(**"СПА-салон 'Релакс'"**,** SwingConstants**.**CENTER**);**

titleLabel**.**setFont**(new** Font**(**"Serif"**,** Font**.**BOLD**,** 36**));**

titleLabel**.**setForeground**(**Color**.**BLUE**);**

JPanel titlePanel **=** **new** JPanel**();**

titlePanel**.**setLayout**(new** BorderLayout**());**

titlePanel**.**add**(**titleLabel**,** BorderLayout**.**CENTER**);**

titlePanel**.**setPreferredSize**(new** Dimension**(**1000**,** 80**));** // Отступ от верха

titlePanel**.**setBackground**(**Color**.**decode**(**"#FFEFD5"**));**

panel**.**add**(**titlePanel**,** BorderLayout**.**NORTH**);**

- JLabel topImageLabel: Отображает изображение в верхней части вкладки.

- JLabel titleLabel: Заголовок вкладки с текстом "СПА-салон 'Релакс'".

- JPanel titlePanel: Панель для размещения заголовка.

1. Левая панель (Выбор филиала и массажа):

Левая панель содержит выпадающий список для выбора филиала и радиокнопки для выбора типа массажа.

// Левая панель: выбор филиала и массажа

JPanel leftPanel **=** **new** JPanel**(new** GridBagLayout**());**

leftPanel**.**setBackground**(**Color**.**decode**(**"#FFEFD5"**));**

GridBagConstraints gbc **=** **new** GridBagConstraints**();**

gbc**.**insets **=** **new** Insets**(**10**,** 10**,** 10**,** 10**);**

gbc**.**fill **=** GridBagConstraints**.**HORIZONTAL**;**

// Выбор филиала

String**[]** branches **=** **{**"Выберите филиал"**,** "ул. Ленина, 10"**,** "пр. Мира, 25"**,** "ул. Советская, 50"**,** "ул. Пушкина, 15"**,** "пр. Гагарина, 30"**};**

branchComboBox **=** **new** JComboBox**<>(**branches**);**

branchComboBox**.**setFont**(new** Font**(**"Serif"**,** Font**.**PLAIN**,** 16**));** // Устанавливаем шрифт

JLabel branchLabel **=** **new** JLabel**(**"Филиал:"**,** SwingConstants**.**RIGHT**);**

branchLabel**.**setFont**(new** Font**(**"Serif"**,** Font**.**PLAIN**,** 16**));** // Устанавливаем шрифт

gbc**.**gridx **=** 0**;** gbc**.**gridy **=** 0**;**

leftPanel**.**add**(**branchLabel**,** gbc**);**

gbc**.**gridx **=** 1**;**

leftPanel**.**add**(**branchComboBox**,** gbc**);**

// Выбор массажа

String**[]** massageTypes **=** **{**"Классический"**,** "Тайский"**,** "Спортивный"**,** "Расслабляющий"**,** "Антицеллюлитный"**};**

massageOptions **=** **new** JRadioButton**[**massageTypes**.**length**];**

ButtonGroup massageGroup **=** **new** ButtonGroup**();**

JLabel massageLabel **=** **new** JLabel**(**"Тип массажа:"**,** SwingConstants**.**RIGHT**);**

massageLabel**.**setFont**(new** Font**(**"Serif"**,** Font**.**PLAIN**,** 16**));** // Устанавливаем шрифт

gbc**.**gridx **=** 0**;** gbc**.**gridy **=** 1**;**

leftPanel**.**add**(**massageLabel**,** gbc**);**

**for** **(**int i **=** 0**;** i **<** massageTypes**.**length**;** i**++)** **{**

massageOptions**[**i**]** **=** **new** JRadioButton**(**massageTypes**[**i**]);**

massageOptions**[**i**].**setFont**(new** Font**(**"Serif"**,** Font**.**PLAIN**,** 16**));** // Устанавливаем шрифт

massageGroup**.**add**(**massageOptions**[**i**]);**

gbc**.**gridx **=** 1**;** gbc**.**gridy **=** i **+** 1**;**

leftPanel**.**add**(**massageOptions**[**i**],** gbc**);**

int finalI **=** i**;**

massageOptions**[**i**].**addActionListener**(**e **->** updateDescriptionAndPrice**(**finalI**));**

**}**

- branchComboBox: Выпадающий список для выбора филиала.

- massageOptions: Радиокнопки для выбора типа массажа.

- updateDescriptionAndPrice(finalI): Метод, который обновляет описание и цену при выборе типа массажа.

1. Правая панель (Описание и изображение):

Правая панель отображает описание выбранного массажа и соответствующее изображение.

// Правая панель: описание и изображение

rightPanel **=** **new** JPanel**(new** BorderLayout**());**

rightPanel**.**setBackground**(**Color**.**decode**(**"#FFEFD5"**));**

descriptionArea **=** **new** JTextArea**();**

descriptionArea**.**setEditable**(false);**

descriptionArea**.**setLineWrap**(true);**

descriptionArea**.**setWrapStyleWord**(true);**

descriptionArea**.**setFont**(new** Font**(**"Serif"**,** Font**.**PLAIN**,** 16**));** // Устанавливаем шрифт

JScrollPane scrollPane **=** **new** JScrollPane**(**descriptionArea**);**

scrollPane**.**setPreferredSize**(new** Dimension**(**300**,** 300**));** // Размер описания

descriptionWrapper **=** **new** JPanel**(new** BorderLayout**());**

descriptionWrapper**.**setBorder**(**BorderFactory**.**createEmptyBorder**(**100**,** 20**,** 0**,** 50**));** // Отступ сверху 100, справа 50

descriptionWrapper**.**setBackground**(**Color**.**decode**(**"#FFEFD5"**));**

descriptionWrapper**.**add**(**scrollPane**,** BorderLayout**.**CENTER**);**

imageLabel **=** **new** JLabel**();**

imageLabel**.**setHorizontalAlignment**(**JLabel**.**CENTER**);**

imageWrapper **=** **new** JPanel**(new** BorderLayout**());**

imageWrapper**.**setBorder**(**BorderFactory**.**createEmptyBorder**(**0**,** 20**,** 50**,** 50**));** // Отступ снизу 50, справа 50

imageWrapper**.**setBackground**(**Color**.**decode**(**"#FFEFD5"**));**

imageWrapper**.**add**(**imageLabel**,** BorderLayout**.**CENTER**);**

- descriptionArea: Текстовое поле для отображения описания массажа.

- imageLabel: Метка для отображения изображения, соответствующего выбранному массажу.

1. Кнопки на первой вкладке:

На первой вкладке есть две кнопки: "Записаться" и "Посмотреть записи".

// Кнопки

bookButton **=** **new** JButton**(**"Записаться"**);**

bookButton**.**setFont**(new** Font**(**"Serif"**,** Font**.**PLAIN**,** 16**));** // Устанавливаем шрифт

bookButton**.**addActionListener**(**e **->** transferSelectionToSecondTab**());**

viewRecordsButton **=** **new** JButton**(**"Посмотреть записи"**);**

viewRecordsButton**.**setFont**(new** Font**(**"Serif"**,** Font**.**PLAIN**,** 16**));** // Устанавливаем шрифт

viewRecordsButton**.**addActionListener**(**e **->** tabbedPane**.**setSelectedIndex**(**2**));**

JPanel buttonPanel **=** **new** JPanel**(new** GridLayout**(**2**,** 1**));**

buttonPanel**.**add**(**bookButton**);**

buttonPanel**.**add**(**viewRecordsButton**);**

gbc**.**gridx **=** 1**;** gbc**.**gridy **=** 8**;**

gbc**.**anchor **=** GridBagConstraints**.**EAST**;** // Выравнивание по правой стороне

leftPanel**.**add**(**buttonPanel**,** gbc**);**

- bookButton: Кнопка "Записаться", которая переносит пользователя на вторую вкладку.

- viewRecordsButton: Кнопка "Посмотреть записи", которая переносит пользователя на третью вкладку.

1. Вторая вкладка (Окончательная запись):

На этой вкладке пользователь подтверждает выбор, указывает дополнительные услуги, телефон и время записи.

private JPanel createSecondTab**()** **{**

JPanel panel **=** **new** JPanel**(new** BorderLayout**())** **{**

@Override

protected void paintComponent**(**Graphics g**)** **{**

**super.**paintComponent**(**g**);**

ImageIcon background **=** **new** ImageIcon**(**"D:/вуз/4 курс/Кетько/курсовая/5.jpg"**);**

g**.**drawImage**(**background**.**getImage**(),** 0**,** 0**,** getWidth**(),** getHeight**(),** **this);** // Рисуем фон без прозрачности

**}**

**};**

panel**.**setBackground**(**Color**.**decode**(**"#FFEFD5"**));**

// Белый прямоугольник для формы записи

JPanel contentPanel **=** **new** JPanel**(new** GridBagLayout**());**

contentPanel**.**setOpaque**(true);**

contentPanel**.**setBackground**(**Color**.**WHITE**);** // Белый фон

contentPanel**.**setPreferredSize**(new** Dimension**(**480**,** 460**));** // Устанавливаем размер

contentPanel**.**setBorder**(**BorderFactory**.**createLineBorder**(**Color**.**BLACK**,** 2**));** // Добавляем границу

- contentPanel: Панель с белым фоном, на которой размещаются все элементы формы записи.

- ImageIcon background: Фоновая картинка для вкладки.

1. Третья вкладка (Мои записи):

На этой вкладке отображается таблица с существующими записями, и пользователь может удалить запись.

private JPanel createThirdTab**()** **{**

JPanel panel **=** **new** JPanel**(new** BorderLayout**());**

panel**.**setBackground**(**Color**.**decode**(**"#FFEFD5"**));**

// Добавляем столбец "Стоимость"

tableModel**.**setColumnIdentifiers**(new** String**[]{**"Филиал"**,** "Массаж"**,** "Доп. услуги"**,** "Время"**,** "Телефон"**,** "Стоимость"**});**

recordsTable **=** **new** JTable**(**tableModel**);**

recordsTable**.**setFont**(new** Font**(**"Serif"**,** Font**.**PLAIN**,** 18**));** // Увеличиваем шрифт

recordsTable**.**setRowHeight**(**30**);** // Увеличиваем высоту строк

recordsTable**.**setBackground**(**Color**.**decode**(**"#FFEFD5"**));**

JScrollPane scrollPane **=** **new** JScrollPane**(**recordsTable**);**

panel**.**add**(**scrollPane**,** BorderLayout**.**CENTER**);**

JButton deleteButton **=** **new** JButton**(**"Удалить запись"**);**

deleteButton**.**setFont**(new** Font**(**"Serif"**,** Font**.**PLAIN**,** 18**));** // Увеличиваем шрифт

deleteButton**.**addActionListener**(**e **->** deleteSelectedRecord**());**

panel**.**add**(**deleteButton**,** BorderLayout**.**SOUTH**);**

**return** panel**;**

**}**

- recordsTable: Таблица для отображения записей.

- deleteButton: Кнопка для удаления выбранной записи.

Приложение предоставляет удобный интерфейс для записи на массаж в СПА-салоне. Оно использует библиотеку Swing для создания графического интерфейса и позволяет пользователю выбирать филиал, тип массажа, дополнительные услуги, а также просматривать и удалять свои записи. Все данные сохраняются в текстовый файл, что обеспечивает их сохранность между запусками приложения.

Описание остальных методов

В приложении используется несколько методов для обработки данных, обновления интерфейса и взаимодействия с пользователем. Рассмотрим каждый из них подробно.

1. Метод `updateDescriptionAndPrice(int index)`:

Этот метод обновляет описание и цену выбранного массажа, а также отображает соответствующее изображение.

private void updateDescriptionAndPrice**(**int index**)** **{**

String**[]** descriptions **=** **{**

"Классический массаж — это базовый вид массажа..."**,**

"Тайский массаж — древняя техника..."**,**

"Спортивный массаж разработан специально для спортсменов..."**,**

"Расслабляющий массаж создан для тех, кто хочет полностью отдохнуть..."**,**

"Антицеллюлитный массаж направлен на борьбу с целлюлитом..."

**};**

int**[]** prices **=** **{**1500**,** 2000**,** 1800**,** 1600**,** 2200**};**

String**[]** imagePaths **=** **{**

"D:/вуз/4 курс/курсовая/1.jpg"**,**

"D:/вуз/4 курс/курсовая/2.jpg"**,**

"D:/вуз/4 курс/курсовая/3.jpg"**,**

"D:/вуз/4 курс/курсовая/4.jpg"**,**

"D:/вуз/4 курс/курсовая/6.jpg"

**};**

**if** **(**index **>=** 0**)** **{**

descriptionArea**.**setText**(**descriptions**[**index**]);**

priceLabel**.**setText**(**"Цена: " **+** prices**[**index**]** **+** " руб."**);**

ImageIcon icon **=** **new** ImageIcon**(**imagePaths**[**index**]);**

Image img **=** icon**.**getImage**().**getScaledInstance**(**600**,** 400**,** Image**.**SCALE\_SMOOTH**);**

imageLabel**.**setIcon**(new** ImageIcon**(**img**));**

rightPanel**.**removeAll**();**

rightPanel**.**add**(**descriptionWrapper**,** BorderLayout**.**NORTH**);**

rightPanel**.**add**(**imageWrapper**,** BorderLayout**.**CENTER**);**

**}** **else** **{**

rightPanel**.**removeAll**();**

rightPanel**.**add**(**initialImageLabel**,** BorderLayout**.**CENTER**);**

**}**

rightPanel**.**revalidate**();**

rightPanel**.**repaint**();**

**}**

- descriptions: Массив строк с описаниями каждого типа массажа.

- prices: Массив цен для каждого типа массажа.

- imagePaths: Массив путей к изображениям, соответствующим каждому типу массажа.

- descriptionArea.setText(descriptions[index]): Устанавливает текст описания в текстовое поле.

- priceLabel.setText(...): Обновляет цену выбранного массажа.

- imageLabel.setIcon(...): Устанавливает изображение для выбранного массажа.

- rightPanel.revalidate() и rightPanel.repaint(): Обновляют панель для отображения изменений.

1. Метод `transferSelectionToSecondTab()`:

Этот метод переносит выбранные данные (филиал и тип массажа) на вторую вкладку.

private void transferSelectionToSecondTab**()** **{**

**if** **(**branchComboBox**.**getSelectedIndex**()** **>** 0 **&&** getSelectedMassage**()** **!=** **null)** **{**

selectedBranchField**.**setText**(**branchComboBox**.**getSelectedItem**().**toString**());**

selectedMassageField**.**setText**(**getSelectedMassage**());**

int basePrice **=** Integer**.**parseInt**(**priceLabel**.**getText**().**replace**(**"Цена: "**,** ""**).**replace**(**" руб."**,** ""**));**

totalPriceField**.**setText**(**basePrice **+** " руб."**);**

tabbedPane**.**setSelectedIndex**(**1**);**

**}** **else** **{**

JOptionPane**.**showMessageDialog**(this,** "Выберите филиал и массаж!"**);**

**}**

**}**

- branchComboBox.getSelectedIndex() > 0: Проверяет, выбран ли филиал (первый элемент в списке — "Выберите филиал").

- getSelectedMassage(): Возвращает выбранный тип массажа.

- selectedBranchField.setText(...): Устанавливает выбранный филиал в текстовое поле на второй вкладке.

- selectedMassageField.setText(...): Устанавливает выбранный тип массажа в текстовое поле на второй вкладке.

- totalPriceField.setText(...): Устанавливает начальную цену массажа.

- tabbedPane.setSelectedIndex(1): Переключает на вторую вкладку.

1. Метод `updateTotalPrice()`:

Этот метод обновляет общую стоимость с учетом дополнительных услуг.

private void updateTotalPrice**()** **{**

**if** **(**basePrice **==** 0**)**

basePrice **=** Integer**.**parseInt**(**totalPriceField**.**getText**().**replace**(**" руб."**,** ""**));**

int additionalPrice1 **=** 0**;**

int additionalPrice2 **=** 0**;**

int additionalPrice3 **=** 0**;**

**if** **(**extraService1**.**isSelected**())** additionalPrice1 **+=** 300**;**

**else** additionalPrice1 **=** 0**;**

**if** **(**extraService2**.**isSelected**())** additionalPrice2 **+=** 300**;**

**else** additionalPrice2 **=** 0**;**

**if** **(**extraService3**.**isSelected**())** additionalPrice3 **+=** 500**;**

elseadditionalPrice3 **=** 0**;**

int currentPrice **=** basePrice **+** additionalPrice1 **+** additionalPrice2 **+** additionalPrice3**;**

totalPriceField**.**setText**(**currentPrice **+** " руб."**);**

**}**

- basePrice: Базовая цена выбранного массажа.

- additionalPrice1, additionalPrice2, additionalPrice3: Стоимость дополнительных услуг (ароматерапия, тепловая терапия, чайная церемония).

- totalPriceField.setText(...): Обновляет общую стоимость в текстовом поле.

1. Метод `saveBooking()`:

Этот метод сохраняет запись в таблицу и файл.

private void saveBooking**()** **{**

String branch **=** selectedBranchField**.**getText**();**

String massage **=** selectedMassageField**.**getText**();**

String extras **=** **(**extraService1**.**isSelected**()** **?** "Ароматерапия " **:** ""**)** **+**

**(**extraService2**.**isSelected**()** **?** "Тепловая терапия " **:** ""**)** **+**

**(**extraService3**.**isSelected**()** **?** "Чайная церемония " **:** ""**);**

String price **=** totalPriceField**.**getText**();**

String phone **=** phoneField**.**getText**();**

String time **=** **(**String**)** timeComboBox**.**getSelectedItem**();**

tableModel**.**addRow**(new** Object**[]{**branch**,** massage**,** extras**,** time**,** price**,** phone**});**

saveRecordsToFile**();**

**}**

- branch: Выбранный филиал.

- massage: Выбранный тип массажа.

- extras: Дополнительные услуги.

- price: Общая стоимость.

- phone: Номер телефона.

- time: Выбранное время записи.

- tableModel.addRow(...): Добавляет запись в таблицу.

- saveRecordsToFile(): Сохраняет все записи в файл.

1. Метод `getSelectedMassage()`:

Этот метод возвращает выбранный тип массажа.

private String getSelectedMassage**()** **{**

**for** **(**int i **=** 0**;** i **<** massageOptions**.**length**;** i**++)** **{**

**if** **(**massageOptions**[**i**].**isSelected**())** **{**

**return** massageOptions**[**i**].**getText**();**

**}**

**}**

**return** **null;**

**}**

- massageOptions[i].isSelected(): Проверяет, выбран ли конкретный тип массажа.

- massageOptions[i].getText(): Возвращает текст выбранного типа массажа.

1. Метод `clearSelection()`:

Этот метод очищает выбор дополнительных услуг, телефон и время записи.

private void clearSelection**()** **{**

extraService1**.**setSelected**(false);**

extraService2**.**setSelected**(false);**

extraService3**.**setSelected**(false);**

phoneField**.**setValue**(null);**

timeComboBox**.**setSelectedIndex**(**0**);**

updateTotalPrice**();**

**}**

- extraService1.setSelected(false): Снимает выбор с дополнительных услуг.

- phoneField.setValue(null): Очищает поле для ввода телефона.

- timeComboBox.setSelectedIndex(0): Сбрасывает выбор времени.

- updateTotalPrice(): Обновляет общую стоимость.

1. Метод `confirmBooking()`:

Этот метод подтверждает запись и сохраняет её.

private void confirmBooking**()** **{**

String message **=** "Вы выбрали:\n" **+**

"Филиал: " **+** selectedBranchField**.**getText**()** **+** "\n" **+**

"Массаж: " **+** selectedMassageField**.**getText**()** **+** "\n" **+**

"Доп. услуги: " **+** **(**extraService1**.**isSelected**()** **?** "Ароматерапия" **:** ""**)** **+ (**extraService2**.**isSelected**()** **?** "Тепловая терапия " **:** ""**)** **+ (**extraService3**.**isSelected**()** **?** "Чайная церемония " **:** ""**)** **+** "\n" **+**

"Цена: " **+** totalPriceField**.**getText**()** **+** "\n" **+**

"Телефон: " **+** phoneField**.**getText**()** **+** "\n" **+**

"Время: " **+** timeComboBox**.**getSelectedItem**()** **+** "\n\n" **+**

"Подтвердить запись?"**;**

int response **=** JOptionPane**.**showConfirmDialog**(this,** message**,** "Подтверждение"**,** JOptionPane**.**YES\_NO\_OPTION**);**

**if** **(**response **==** JOptionPane**.**YES\_OPTION**)** **{**

saveBooking**();**

JOptionPane**.**showMessageDialog**(this,** "Вы успешно записаны. С вами свяжутся по предоставленному " **+**

"номеру телефона. \nСпасибо за выбор 'Релакс'!"**);**

**}**

**}**

- message: Сообщение с подтверждением выбора.

- JOptionPane.showConfirmDialog(...): Отображает диалоговое окно с подтверждением.

- saveBooking(): Сохраняет запись, если пользователь подтвердил.

1. Метод `loadRecords()`:

Этот метод загружает записи из файла в таблицу.

private void loadRecords**()** **{**

tableModel**.**setRowCount**(**0**);** // Очистка таблицы перед загрузкой

File file **=** **new** File**(**FILE\_PATH**);**

**if** **(!**file**.**exists**())** **{**

**try** **{**

file**.**createNewFile**();** // Создание файла, если его нет

**}** **catch** **(**IOException e**)** **{**

e**.**printStackTrace**();**

**}**

**}**

**try** **(**BufferedReader reader **=** **new** BufferedReader**(new** FileReader**(**file**)))** **{**

String line**;**

**while** **((**line **=** reader**.**readLine**())** **!=** **null)** **{**

String**[]** data **=** line**.**split**(**";"**);**

**if** **(**data**.**length **==** tableModel**.**getColumnCount**())** **{**

tableModel**.**addRow**(**data**);**

**}**

**}**

**}** **catch** **(**IOException e**)** **{**

e**.**printStackTrace**();**

**}**

**}**

- tableModel.setRowCount(0): Очищает таблицу перед загрузкой новых данных.

- BufferedReader reader: Читает данные из файла.

- tableModel.addRow(data): Добавляет строку в таблицу.

1. Метод `deleteSelectedRecord()`:

Этот метод удаляет выбранную запись из таблицы и файла.

private void deleteSelectedRecord**()** **{**

int selectedRow **=** recordsTable**.**getSelectedRow**();**

**if** **(**selectedRow **!=** **-**1**)** **{**

int response **=** JOptionPane**.**showConfirmDialog**(this,** "Вы уверены, что хотите удалить эту запись?"**,** "Подтверждение"**,** JOptionPane**.**YES\_NO\_OPTION**);**

**if** **(**response **==** JOptionPane**.**YES\_OPTION**)** **{**

tableModel**.**removeRow**(**selectedRow**);**

saveRecordsToFile**();** // Обновляем файл после удаления

**}**

**}** **else** **{**

JOptionPane**.**showMessageDialog**(this,** "Выберите запись для удаления!"**);**

**}**

**}**

- recordsTable.getSelectedRow(): Возвращает индекс выбранной строки.

- tableModel.removeRow(selectedRow): Удаляет строку из таблицы.

- saveRecordsToFile(): Сохраняет изменения в файл.

1. Метод `saveRecordsToFile()`:

Этот метод сохраняет все записи из таблицы в файл.

private void saveRecordsToFile**()** **{**

**try** **(**BufferedWriter writer **=** **new** BufferedWriter**(new** FileWriter**(**FILE\_PATH**)))** **{**

**for** **(**int i **=** 0**;** i **<** tableModel**.**getRowCount**();** i**++)** **{**

StringBuilder line **=** **new** StringBuilder**();**

**for** **(**int j **=** 0**;** j **<** tableModel**.**getColumnCount**();** j**++)** **{**

line**.**append**(**tableModel**.**getValueAt**(**i**,** j**)).**append**(**";"**);**

**}**

writer**.**write**(**line**.**toString**().**strip**()** **+** "\n"**);**

**}**

**}** **catch** **(**IOException e**)** **{**

e**.**printStackTrace**();**

**}**

**}**

- BufferedWriter writer: Записывает данные в файл.

- tableModel.getValueAt(i, j): Получает значение ячейки таблицы.

- writer.write(...): Записывает строку в файл.

Эти методы обеспечивают функциональность приложения, включая выбор услуг, обновление интерфейса, сохранение и загрузку данных, а также удаление записей. Каждый метод выполняет конкретную задачу, что делает код модульным и легко поддерживаемым.

* 1. Экранные формы

Главное окно приложения при открытии представлено на рисунке 6.

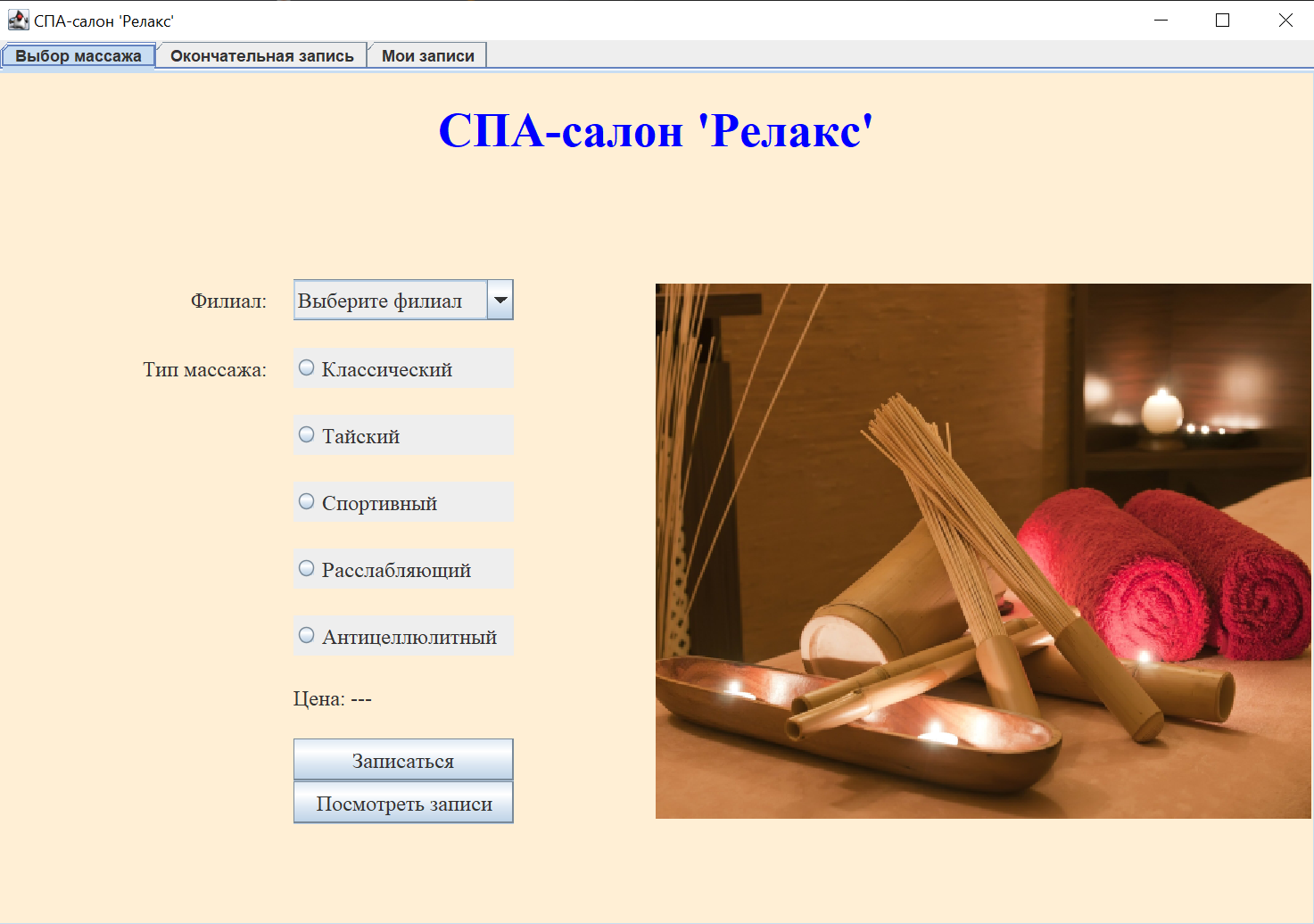


Рисунок 6 - Главное окно приложения при открытии

Отображение списка выбора филиала представлено на рисунке 7.

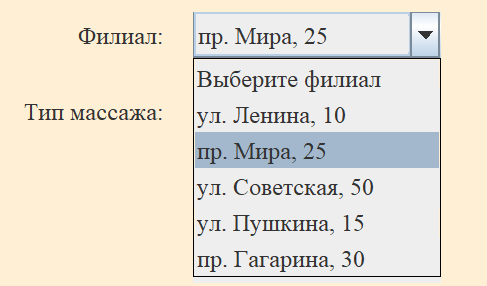


Рисунок 7 - Отображение списка филиалов

Отображение смены стартового изображения на описание типа массажа и его изображения представлено на рисунке 8. Поле «Цена» меняется в зависимости от выбранного типа массажа.



Рисунок 8 - Описание типа массажа и его изображение

После нажатия на кнопку «Записаться» открывается вкладка «Окончательная запись». Вкладка представлена на рисунке 9.

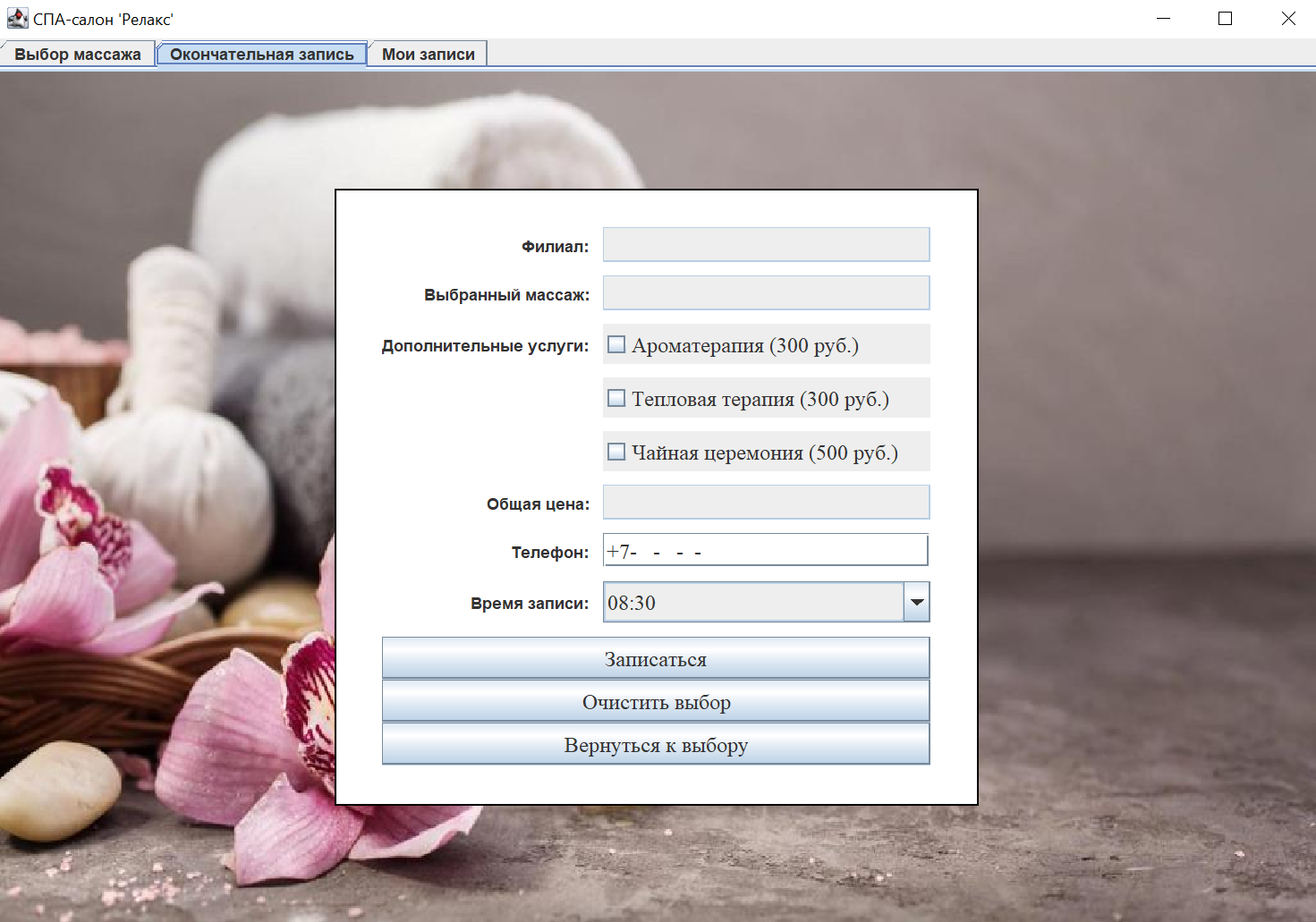


Рисунок 9 - Вкладка "Окончательная запись"

При выборе доп. услуг, цена автоматически обновляется. Заполненная форма записи представлена на рисунке 10.

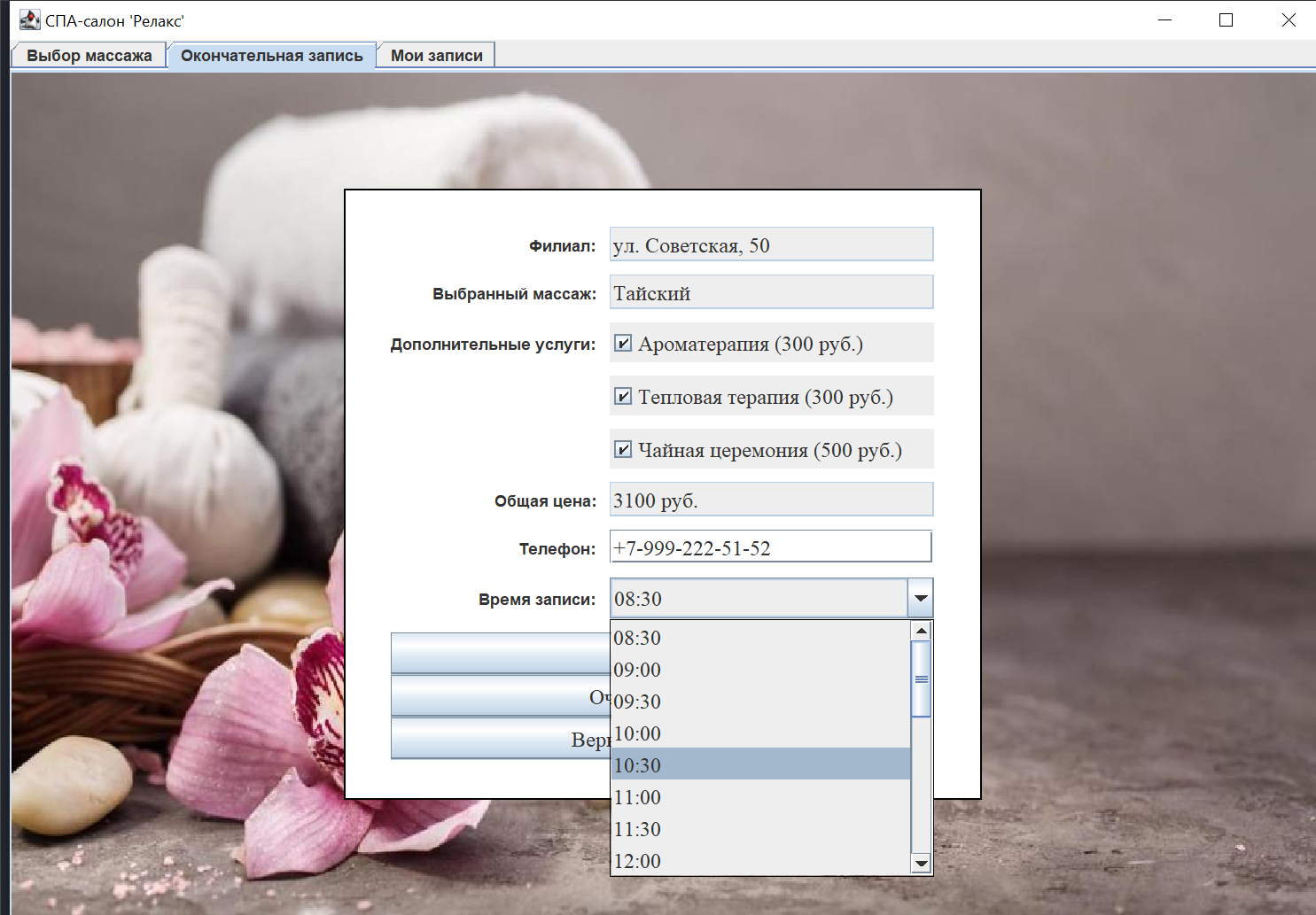


Рисунок 10 - Заполненная форма записи

После выбора доп. услуг, времени записи и указания номера телефона, пользователь нажимает на кнопку «Записаться». Всплывающее окно проверки заполненной информации представлено на рисунке 11.

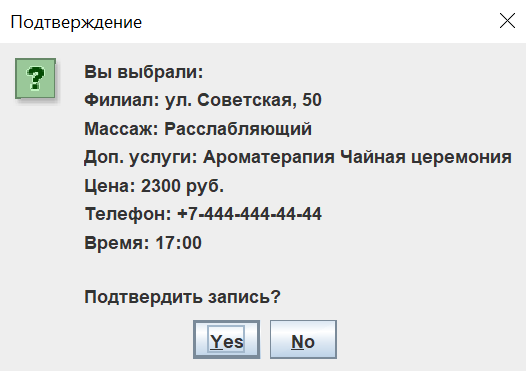


Рисунок 11 - Всплывающее окно проверки информации

Также на рисунке 10 представлена кнопка «Очистить выбор». При ее нажатии заполняемые поля формы записи стираются и возвращаются к исходному состоянию, как на рисунке 9.

При нажатии «Yes» на окне подтверждения (рисунок 11), появляется всплывающее окно, подтверждающее успешную запись. Окно представлено на рисунке 12.

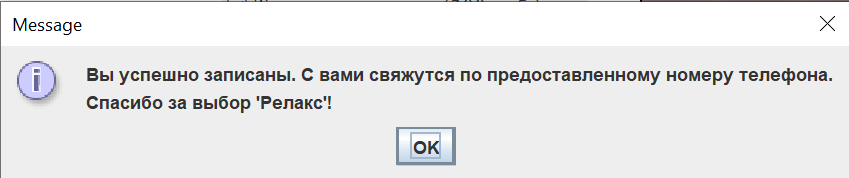


Рисунок 12 - Всплывающее окно подтверждения записи

На третьей вкладке пользователь может посмотреть созданные им записи. Вкладка «Мои записи» представлена на рисунке 13.

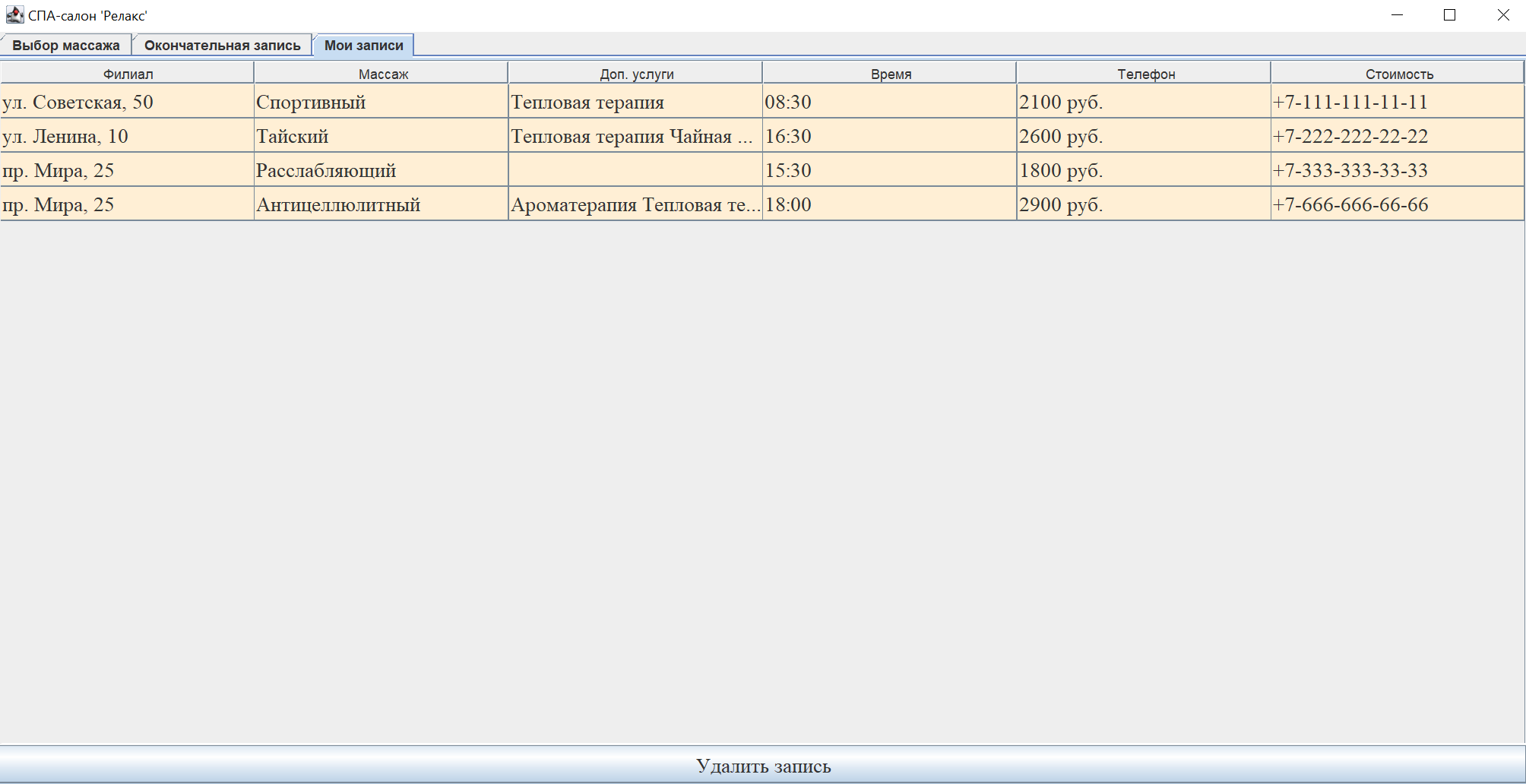


Рисунок 13 - Вкладка "Мои записи"

Любая запись из таблицы может быть удалена. Для этого надо выбрать запись и нажать кнопку «Удалить запись». После нажатия кнопки появляется всплывающее окно подтверждения удаления. Окно подтверждения представлено на рисунке 14.

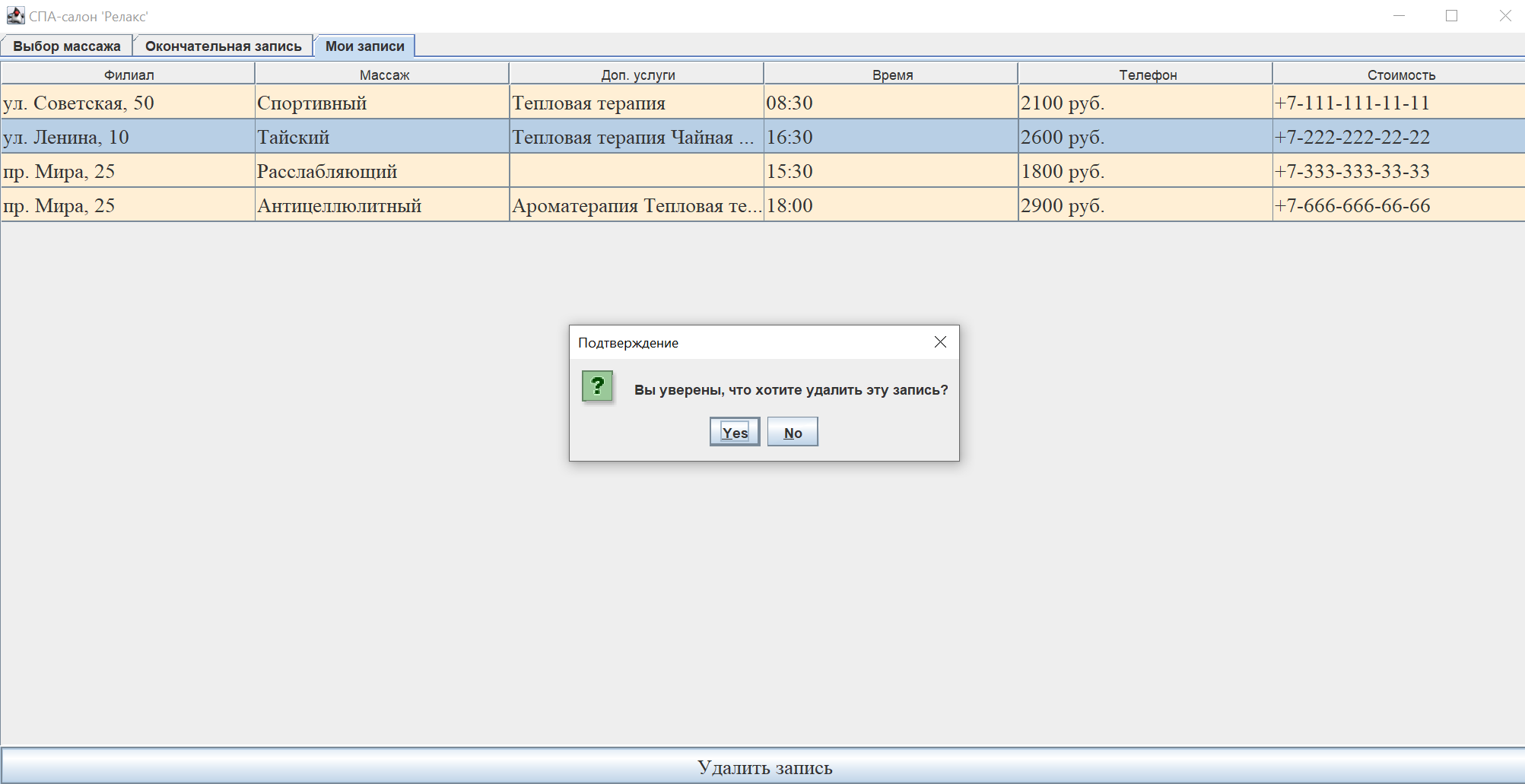


Рисунок 14 - Всплывающее окно подтверждения удаления записи

При нажатии кнопки «Yes» запись удаляется.

ЗАКЛЮЧЕНИЕ

Разработка приложения для записи на массаж в СПА-салоне на языке программирования Java с использованием встроенной библиотеки Swing представляет собой автоматизацию услуг, обеспечивая удобство и комфорт для клиентов и администрации салона. Приложение позволяет пользователю выбирать филиал, тип массажа, дополнительные услуги, а также указывать контактные данные и время записи. Для удобства навигации и визуальной организации интерфейса были созданы три вкладки: выбор массажа, окончательная запись и просмотр существующих записей.

В ходе работы были продемонстрированы основные этапы разработки приложения, включая создание графического интерфейса, обработку пользовательских данных и взаимодействие с файловой системой для сохранения и загрузки записей. Были использованы различные компоненты Swing, такие как JTabbedPane, JComboBox, JRadioButton, JCheckBox, JTable и другие, что позволило создать интуитивно понятный и функциональный интерфейс. Особое внимание было уделено обработке событий, что обеспечивает корректное обновление данных и отображение информации в реальном времени.

Разработанное приложение демонстрирует возможность эффективного использования Java Swing для создания программных решений в сфере услуг. Оно предоставляет пользователю возможность легко и быстро записаться на массаж. В дальнейшем приложение может быть усовершенствовано путем добавления новых функций, таких как интеграция с базой данных, расширение списка услуг, улучшение визуального оформления и адаптация под мобильные устройства. Это позволит сделать приложение еще более удобным и востребованным в условиях растущей конкуренции в сфере СПА-услуг.

Таким образом, данная работа подтверждает, что Java Swing является мощным инструментом для создания приложений с графическим интерфейсом, которые могут успешно применяться в различных сферах, включая индустрию красоты и здоровья.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Кетько, Н.В. Проектирование информационных систем: лекции/ ВолгГТУ. – 2024.
2. Давыдовский, М.А. Проектирование программной системы в UML Designer: учебное пособие / М. А. Давыдовский, М. Н. Никольская. – Москва: РУТ (МИИТ), 2019. – 129 с.
3. Флегонтов, А. В. Моделирование информационных систем: учебное пособие / А.В.Флегонтов, И. Ю. Матюшичев. – 2-е издание, стер.
4. Остроух А.В., Суркова Н.Е. Проектирование информационных систем: учебное пособие / А.В. Остроух, Н.Е Суркова. – 2-е издание, стер. – 2021. – 164 с.
5. Барков, И.А. Объектно-ориентированное программирование: учебник / И.А. Барков. – Санкт-Петербург: Лань, 2019. – 700с.