Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Инженерно-экономический факультет

Кафедра экономической информатики

Дисциплина: Распределенные системы обработки информации

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему

РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ БРОНИРОВАНИЯ НОМЕРОВ В ГОСТИНИЦЕ

Студент:  гр.

Руководитель: ассистент кафедры

Минск 2023

**СОДЕРЖАНИЕ**

[Введение 3](#_Toc103206260)

[1 Описание процесса бронирования номера в гостинице через систему интернет-бронирования 5](#_Toc103206261)

[1.1 Описание предметной области 5](#_Toc103206262)

[1.2 Примеры существующих решений на рынке 6](#_Toc103206262)

[1.3 Анализ требований к разрабатываемому программному средству 8](#_Toc103206263)

[2 Разработка функциональной модели системы бронирования номеров в гостинице 11](#_Toc103206264)

[3 Разработка информационной модели автоматизированной системы бронирования номера в гостинице 15](#_Toc103206265)

[4 Проектирование системы бронирования номеров в гостинице на основе uml 20](#_Toc103206267)

[5 Обоснование выбора компонентов и технологий для реализации автоматизированной системы бронирования номеров в гостинице 29](#_Toc103206268)

7 [Описание алгоритмов, реализующих ключевую бизнес-логику ситемы бронирования номеров в гостинице 31](#_Toc103206272)

8[Руководство по развертыванию и использованию системы бронирования номеров в гостинице 28](#_Toc103206273)

[Заключение 33](#_Toc103206274)

[Список использованных источников 34](#_Toc103206275)

[Приложение А 35](#_Toc103206276)

[Приложение Б 38](#_Toc103206277)

[Приложение В 40](#_Toc103206278)

[Приложение Г 45](#_Toc103206279)

[Приложение Д 46](#_Toc103206280)

**ВВЕДЕНИЕ**

Развитие технологий не остановилось на одном лишь создании интернета и мобильных устройств. Сегодня автоматизация затрагивает все сферы жизни человека, включая туризм и гостиничный бизнес. В связи с этим возникает необходимость создания автоматизированной системы бронирования номеров в гостинице.

Автоматизированная система бронирования номеров в гостинице необходима для упрощения процесса бронирования и увеличения эффективности работы гостиницы. Она позволяет клиентам самостоятельно выбирать и бронировать номера, а также оплачивать их онлайн. Это сокращает время, которое клиенты тратят на ожидание ответа от гостиницы и уменьшает количество ошибок в процессе бронирования. Кроме того, автоматизация процесса бронирования позволяет гостинице легче контролировать заполненность номеров, управлять ценами и предложениями, а также снижать затраты на обслуживание гостей.

Первое, что приходит на ум при упоминании бронирования номеров в гостинице – это удобство для клиента. Клиент может забронировать номер, не выходя из дома или офиса, выбрав подходящий вариант из предложенных на сайте гостиницы. Также клиент может ознакомиться с фотографиями номеров и дополнительными услугами, которые предлагает гостиница. Это позволяет клиенту сэкономить время и выбрать наиболее подходящий вариант проживания.

Кроме удобства для клиента, автоматизированная система бронирования номеров в гостинице позволяет упростить работу персонала гостиницы. Она автоматически обрабатывает запросы на бронирование, создает заказы и отправляет уведомления клиентам о подтверждении бронирования. Это значительно снижает нагрузку на персонал гостиницы и позволяет им заниматься более важными задачами.

Еще одним преимуществом автоматизированной системы бронирования номеров в гостинице является возможность увеличить количество бронирований и, соответственно, доходов гостиницы. Клиенты могут забронировать номер в любое время суток, а не только в рабочие часы гостиницы. Это позволяет гостинице привлечь больше клиентов и увеличить свой доход.

Кроме того, автоматизированная система бронирования номеров в гостинице позволяет сократить количество ошибок при обработке заказов. Все данные о бронировании хранятся в единой базе данных, что позволяет избежать ошибок при передаче информации от одного сотрудника к другому. Это уменьшает количество недовольных клиентов и повышает уровень сервиса гостиницы.

Наконец, автоматизированная система бронирования номеров в гостинице позволяет собирать и анализировать данные о клиентах. Гостиница может узнать, какие услуги наиболее популярны среди клиентов, и на основе этой информации разрабатывать новые предложения. Также гостиница может узнать, какие клиенты чаще всего бронируют номера, и предложить им специальные условия проживания.

*Целью курсового проекта* является создание автоматизированной системы бронирования номеров в гостинице для упрощения и автоматизации процесса бронирования номеров для гостей, сокращении времени на обработку заявок и улучшении качества обслуживания.

Поставленная цель потребовала решения *следующих задач*:

* описать процессы интернет-бронирования;
* поставить задачу разработки автоматизированной системы;
* разработать и описать модели представления системы;
* разработать информационную модель системы;
* разработать модуль для управления списком гостиниц и онлайн-броней;
* обосновать выбор архитектурных решений, используемых при разработке;
* описать алгоритмы, реализующие бизнес-логику приложения;
* написать руководство пользователя.

Объектом исследования курсового проекта является система интернет-бронирования отелей, для формирования броней в конкретной стране или городе.

Предметом исследования курсового проекта являются способы и пути повышения качества обслуживания клиентов и сокращение времени на обработку заявок.

|  |  |  |
| --- | --- | --- |
| **1** | **Описание процесса бронирования номера в гостинице через систему интернет-бронирования** | |
| **1.1** | | **Описание предметной области** |

Система интернет-бронирования является автоматизированной системой, которая позволяет гостям забронировать номер в гостинице через интернет. Эта система предоставляет удобный и быстрый способ бронирования номера, который может быть осуществлен из любой точки мира.

Процесс бронирования номера начинается с поиска подходящего номера на сайте гостиницы. Гости могут выбрать различные параметры, такие как дата заезда и выезда, количество гостей, тип номера и другие параметры, чтобы найти подходящий номер. Когда гость находит подходящий номер, он может забронировать его, заполнив форму бронирования на сайте.

Форма бронирования содержит информацию о госте, такую как имя, фамилия, адрес электронной почты и номер телефона. Гость также может указать дополнительные требования, такие как требования к питанию или услугам гостиницы. Когда гость заполняет форму бронирования, он должен также указать данные своей кредитной карты для оплаты бронирования.

Система интернет-бронирования автоматически проверяет наличие свободных номеров на выбранные даты и резервирует номер для гостя. Если номер недоступен на выбранные даты, система предложит другие варианты номеров или даты, которые могут быть доступны.

Когда бронирование подтверждено, гость получает электронное письмо с подтверждением бронирования и информацией о гостинице, включая адрес, контактную информацию и инструкции по заезду. Гость также может получить доступ к своему бронированию через систему интернет-бронирования, чтобы проверить информацию о бронировании, изменить или отменить бронирование.

Система интернет-бронирования также может быть связана с другими системами управления гостиницей, такими как система управления бронированием и система управления номерами. Это позволяет автоматически обновлять информацию о наличии номеров и избежать двойных бронирований.

Кроме того, система интернет-бронирования может предоставлять гостям дополнительные услуги, такие как заказ трансфера из аэропорта или услуги прачечной. Гости могут заказать эти услуги при бронировании номера через систему интернет-бронирования.

Система интернет-бронирования также может предоставлять гостинице ценную информацию о потребностях гостей и популярности различных услуг. Это помогает гостинице улучшить качество своих услуг и повысить удовлетворенность гостей.

В целом, система интернет-бронирования является эффективным и удобным способом бронирования номеров в гостинице для гостей. Она позволяет сократить время на обработку заявок, улучшить качество обслуживания и повысить доходы гостиницы.

|  |  |
| --- | --- |
| **1.2** | **Примеры существующих решений на рынке** |

В наше динамичное время трудно уже представить себе современный отель без системы автоматизации, где процесс оформления поселения и выселения проводится на рецепции вручную, очень медленно, и гости вынуждены тратить время на раздражающее ожидание в очереди. Улучшить качество управления средством размещения и свести человеческий фактор к минимуму помогает программа автоматизации гостиничного бизнеса — PMS (Property Management System).

**Shelter**

Современная гостиничная система автоматизации Shelter v.2 разработки компании UCS позволяет оформить гостя в течение считанных минут, сделать в целом процесс поселения и выселения быстрым и комфортным как для гостей, так и для персонала гостиницы.

Компания UCS является единственным разработчиком ПО, который может на базе собственных программных решений предложить полную автоматизацию всех зон гостиничного комплекса. Информационная система гостиницы Shelter взаимодействует с системой управления рестораном R-Keeper, программой автоматизации фитнес клуба и СПА - Абонемент, складской учетной системой StoreHouse.

**Booking.com**

Cистема [интернет-бронирования](https://ru.wikipedia.org/wiki/%D0%9E%D0%BD%D0%BB%D0%B0%D0%B9%D0%BD-%D0%B1%D1%80%D0%BE%D0%BD%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) [отелей](https://ru.wikipedia.org/wiki/%D0%93%D0%BE%D1%81%D1%82%D0%B8%D0%BD%D0%B8%D1%86%D0%B0). Бизнес-модель очень простая — отели выставляют предложения, пользователи выбирают подходящие и платят отелям, которые раз в месяц выплачивают Букингу комиссию.

Информация о технологическом стеке и языках программирования, используемых в Booking.com, не является общедоступной. Однако известно, что Booking.com использует собственную разработку, которая базируется на микросервисной архитектуре и включает в себя компоненты, написанные на различных языках программирования, включая Java, Python, Ruby и др. Они также используют различные инструменты и технологии, такие как Docker, Kubernetes, Hadoop и другие.

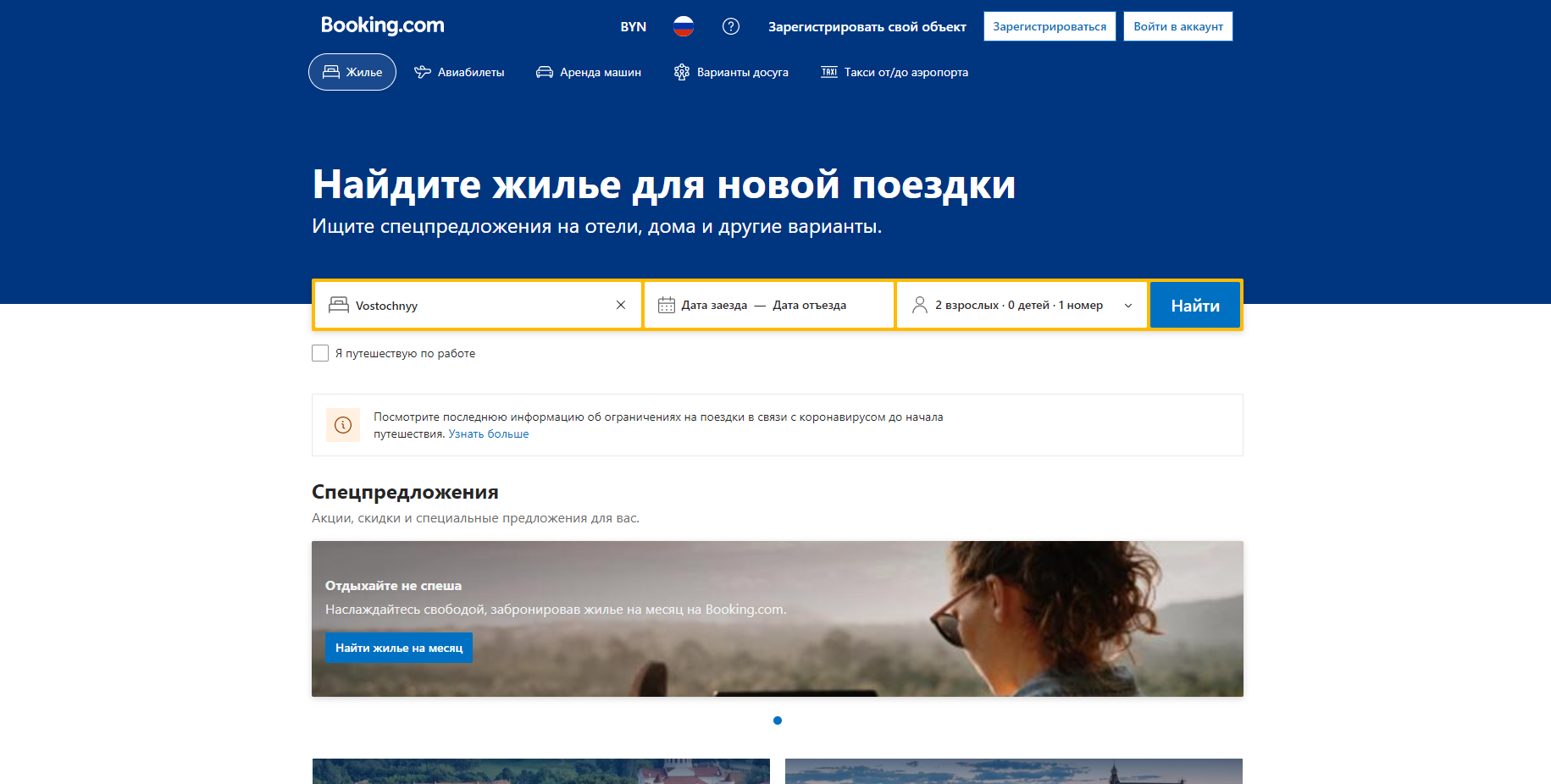


Рисунок 1.1 – Главная страница сайта Booking.com

При поиске отелей в системе Booking, можно установить несколько критериев в фильтре:

* звездность отелей;
* наличие завтраков в отеле;
* номер для некурящих;
* оплата (моментальная) или на месте;
* требуется ли кредитная карта при бронировании или нет;
* бесплатна отмена бронирования.

Сильные стороны: большой выбор отелей по всему миру; простота бронирования; возможность бронирования без кредитной карты; на сайте есть фото отеля, отзывы туристов, а также возможность посмотреть расположение отеле на карте (актуально для пляжного отдыха).

Слабые сторны: фото гостиницы могут быть устаревшие; фиктивные скидки; спам после регистрации.

**Ostrovok.ru** — российский сервис онлайн-бронирования [отелей](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%B5%D0%BB%D1%8C).

Идет большой упор на клиентскую часть, который требует того, чтобы код был тщательно организован и структурирован: выбор модульного подхода к разработке, основанный на ООП.

Это значит, что в системе активно используются классы, наследования, слабое связывание и прочие методы, которые предоставляет JavaScript.

Для каждой страницы есть отдельный JavaScript-класс, содержащий в себе какие-то вложенные модули, которые, в свою очередь, могут дробиться постоянно.

Тщательная организация кода, также касается верстки: серверных и клиентских шаблонов и CSS. Требования к верстке те же самые, что и к JavaScript-приложению: модульность, гибкость, настраиваемость, безболезненное наследование.

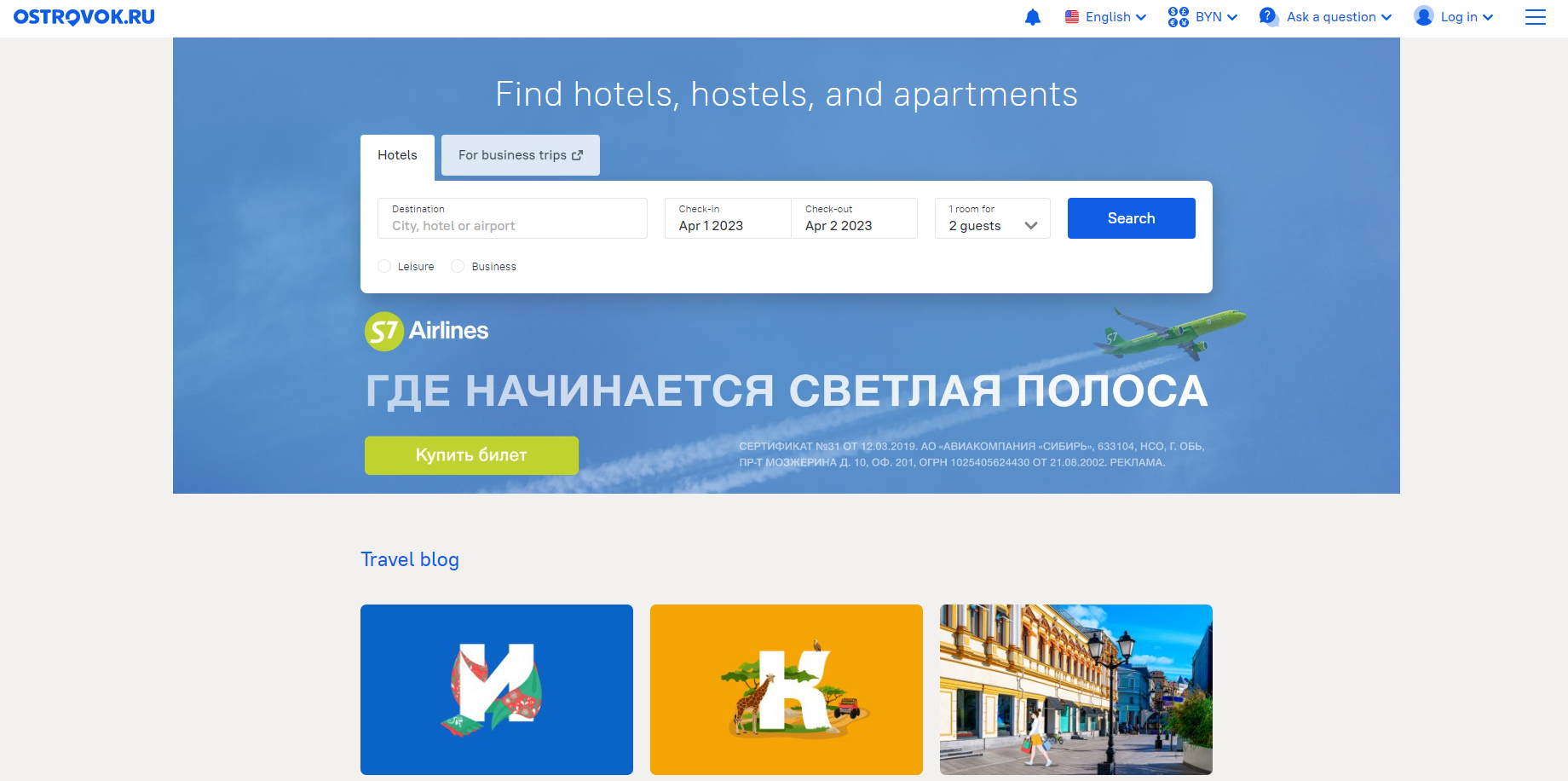
****

Рисунок 1.2 – Главная страница сайта Ostrovok.ru

Плюс, с таким подходом к разработке, код проекта становится гибким и легко приспосабливаемым к нетривиальным ситуациям. Например, если перед разработчиком стоит задача кастомизировать страницу выдачи для пользователей, пришедших на наш сайт по поисковому запросу, а не с главной страницы, то задача ограничивается созданием дочернего класса и переопределением нескольких методов у родительского класса.

|  |  |
| --- | --- |
| **1.3** | **Анализ требований к разрабатываемому программному средству** |

В современном мире гостиничное дело развивается такими темпами, что первостепенной задачей по управлению отелем становятся высокий сервис в гостиничном бизнесе: качественные гостиничные услуги и грамотный гостиничный менеджмент. Гостиничные услуги, предъявляют все больше требований к управлению отелем. Комплексная автоматизация гостиниц, включающая в себя автоматизацию процессов бронирования, расчетов с гостями, агентами, туроператорами и хозяйственной деятельности становятся все более актуальными для предприятий гостиничной индустрии. Современные системы автоматизации гостиниц (PMS системы для гостиниц) включают в себя множество функций, среди которых:

Современные системы автоматизации гостиниц (PMS системы для гостиниц) включают в себя множество функций, среди которых:

* автоматизация баров и ресторанов;
* автоматизация систем управления гостиницей PSM (property management system);
* системы онлайн бронирования гостиниц;
* система безопасности гостиницы;
* системы контроля доступа для гостиниц;
* система управления персоналом гостиницы.

Автоматизированная система бронирования номеров должна удовлетворять следующим требованиям:

1. Онлайн-бронирование: система должна обеспечивать возможность бронирования номеров через интернет.

2. Надежность: система должна быть надежной и стабильной, чтобы пользователи могли бронировать номера без сбоев.

3. Удобство использования: система должна быть простой в использовании, чтобы пользователи могли легко найти нужную информацию и забронировать номер.

4. Гибкость: система должна быть гибкой и адаптивной к различным типам номеров и услуг.

5. Многоплатформенность: система должна работать на различных устройствах, включая компьютеры, мобильные устройства и планшеты.

6. Интеграция: система должна интегрироваться с другими системами, такими как системы управления отелями и платежные системы.

7. Безопасность: система должна обеспечивать безопасность персональных данных пользователей и защиту от мошенничества.

8. Аналитика: система должна предоставлять аналитические данные о бронировании номеров для улучшения управления отелем и увеличения прибыли.

9. Поддержка: система должна иметь круглосуточную техническую поддержку для решения проблем пользователей.

10. Мультиязычность: система должна поддерживать несколько языков для удобства использования пользователей из разных стран.

|  |  |
| --- | --- |
| **2** | **РАЗРАБОТКА ФУНКЦИОНАЛЬНОЙ МОДЕЛИ СИСТЕМЫ БРОНИРОВАНИЯ НОМЕРОВ В ГОСТИНИЦЕ** |

Автоматизированная система бронирования номеров будет представлять собой интернет площадку с возможностью поиска и выбора в первую очередь временного жилья.

Пользователь устанавливает критерии для поиска и выбора жилья и дополнительных жилищных услуг в заданных им самим странах, городах, заказывает авиабилеты для осуществления их бронирования или покупки, оставляет заявку на предоставление услуг. Каждый пользователь обязан будет иметь персональный аккаунт для использования и управления предоставляемых сайтом услуг.

С целью отображения взаимодействия подсистемы с внешним миром построена контекстная диаграмма, которая является вершиной древовидной структуры диаграмм стандарта IDEF0.

IDEF0 обозначает определение интеграции для моделирования процессов — методологию создания общедоступных доменов, которая используется для моделирования предприятий и их процессов, чтобы их можно было понять и улучшить. Схемы IDEF0 (рисунок 2.1) обычно включают следующие компоненты:

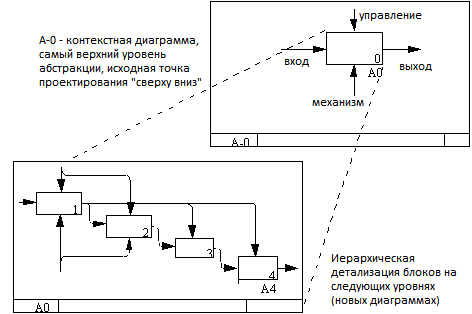
* Контекстная схема — самая верхняя схема в модели IDEF0.
* Иерархия декомпозиции IDEF0 с использованием родительских и детских связей.
* Дерева узлов — структуры узлов в виде дерева, корневые на выбранном узле и используемые для представления полного декомпозиции IDEF0 в одной схеме.

Рисунок 2.1 - Основы графической нотации функционального моделирования бизнес-процессов IDEF0

Стрелка слева – это «вход» (обозначает начало процесса), без которого не может начаться процесс.

Сам процесс будет выполнен в конечный итог — «выход».

Стрелка «механизм» — это то, посредством чего процесс производится. Это могут быть люди или инструменты.

Стрелка «управление» — это управляющее воздействие на процесс, например, инструкция, алгоритм, спецификации.

Подводя итог возможностям и ограничениям IDEF0, еще раз подчеркнем достоинства этой методологии функционального моделирования:

* отлично подходит для иерархического проектирования «сверху вниз», от целых направлений деятельности до локальных бизнес-функций;
* позволяет комплексно представить все бизнес-процессы предприятия в виде единой системы из взаимосвязанных компонентов, отделяя ее внутреннее устройство от внешних объектов/сигналов;
* дает быстрое представление о наиболее важных характеристиках процесса (вход, выход, управление, механизмы), задействованных в его выполнении;
* является хорошим способом идентификации процессов-источников доходов и затрат.

Основную прибыль гостиница в первую очередь получает из бронирования номеров через интернет, располагая тарифами для частных лиц, а также для компаний. Для понимания процесса бронирования номера обеих категорий была воспроизведена функциональная модель в программе AllFusion.

Входные данные в данной модели представляют собой информацию о клиенте и гостиницах: данные клиента, такие как пароль и e-mail, а также имя, фамилия и дата рождения для неавторизованных пользователей.

Для управления процессом бронирования клиента выступают: документ «Политика в отношении обработки персональных данных» и закон Республики Беларусь «О защите персональных данных», которые дают гарантию безопасности и нераспространения данных клиента при предоставлении личной информации. Это необходимо так как при оформлении брони используются паспортные данные клиента.

Для бронирования номера в гостинице необходим пользователь, который является клиентом. Клиент заполняет все формы на сайте, для оформления брони. Также необходим администратор, который будет управлять процессом, держать контроль над пользователями и обрабатывать заявки со своей стороны.

В конечном итоге бронируется номер, а также, после проживания, появляется возможность оставить свой отзыв об оказанной или не оказанной услуге.

Для графического представления процесса бронирования номера в гостинице была представлена диаграмма верхнего уровня модели IDEF0 на рисунке 2.2.

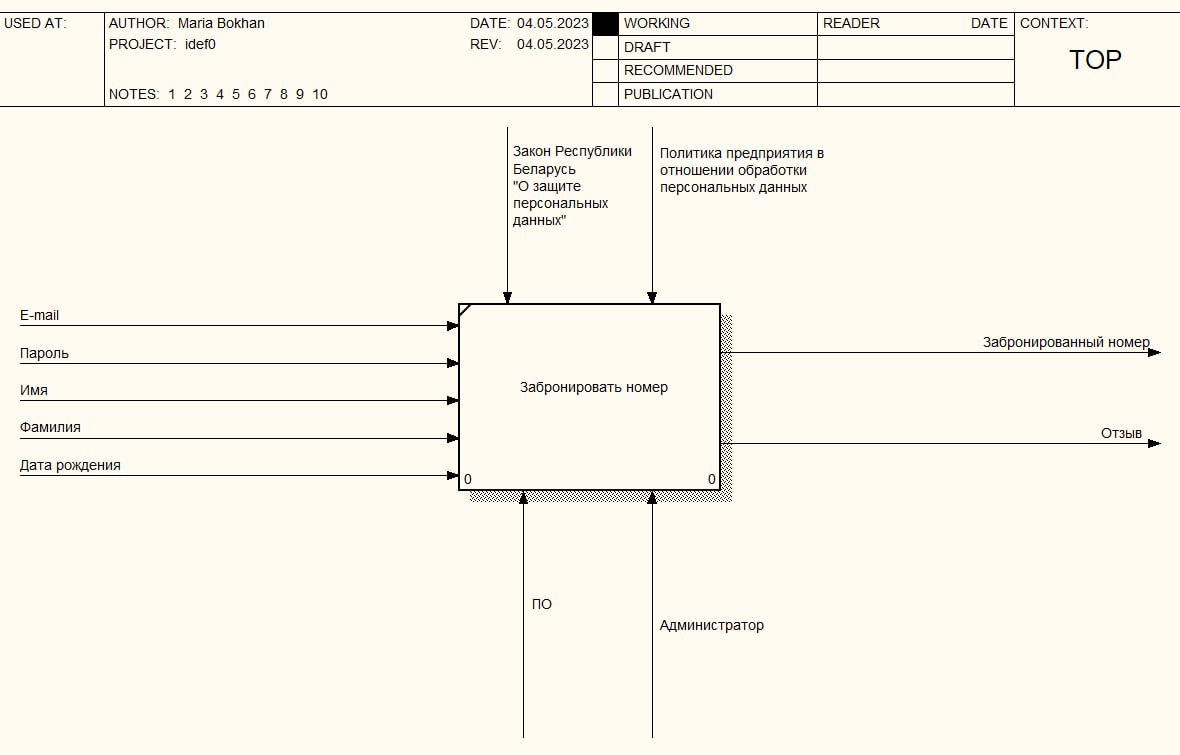
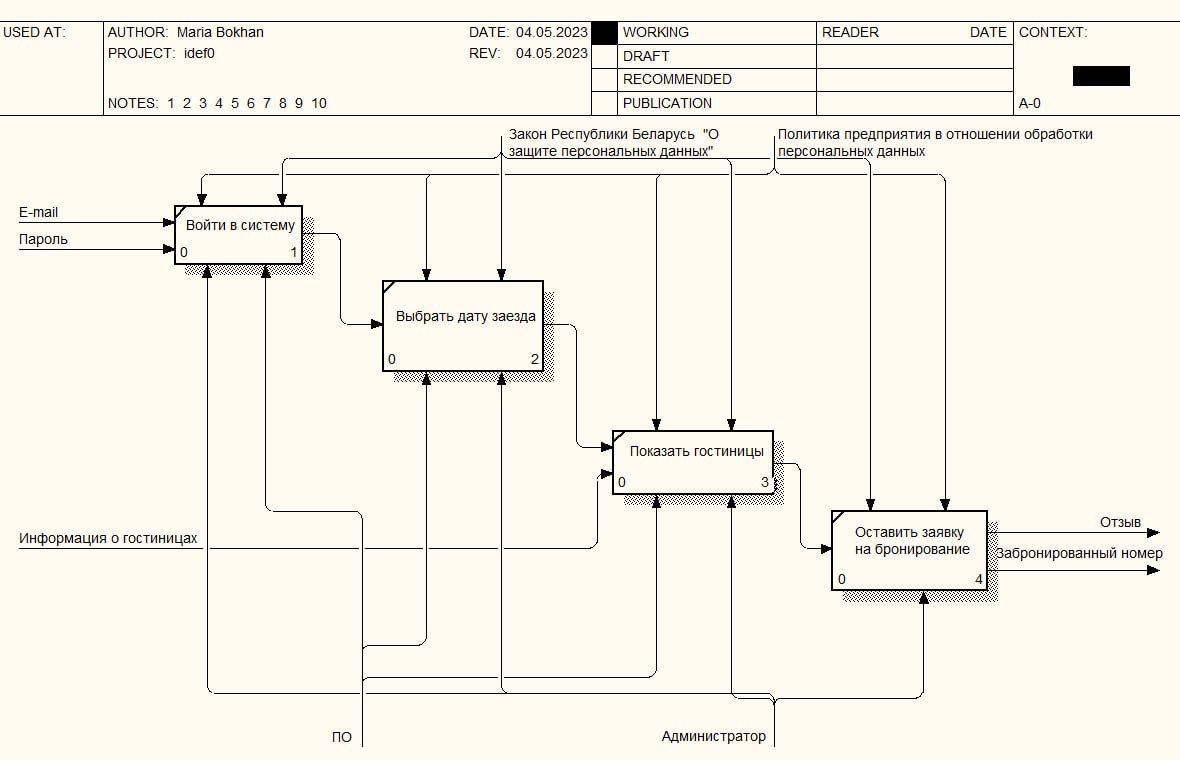


Рисунок 2.2 – Контекстная диаграмма процесса «Забронировать номер в гостинице»

Чтобы забронировать номер, клиенту для начала необходимо зарегистрироваться и выполнить вход в систему. После этого необходимо указать желаемую дату своего заезда и выезда. После этого клиент может ознакомиться со списком подходящих для него гостиниц. После выбора понравившейся гостиницы клиенту необходимо оставить заявку на бронирование на сайте. Обработкой всего процесса управляют программное обеспечение и ответственный администратор. Контроль процесса осуществляется законом государства и политикой предприятия. Графическое представление данного процесса представлено на рисунке 2.3.

Рисунок 2.3 - Декомпозиция диаграммы IDEF0

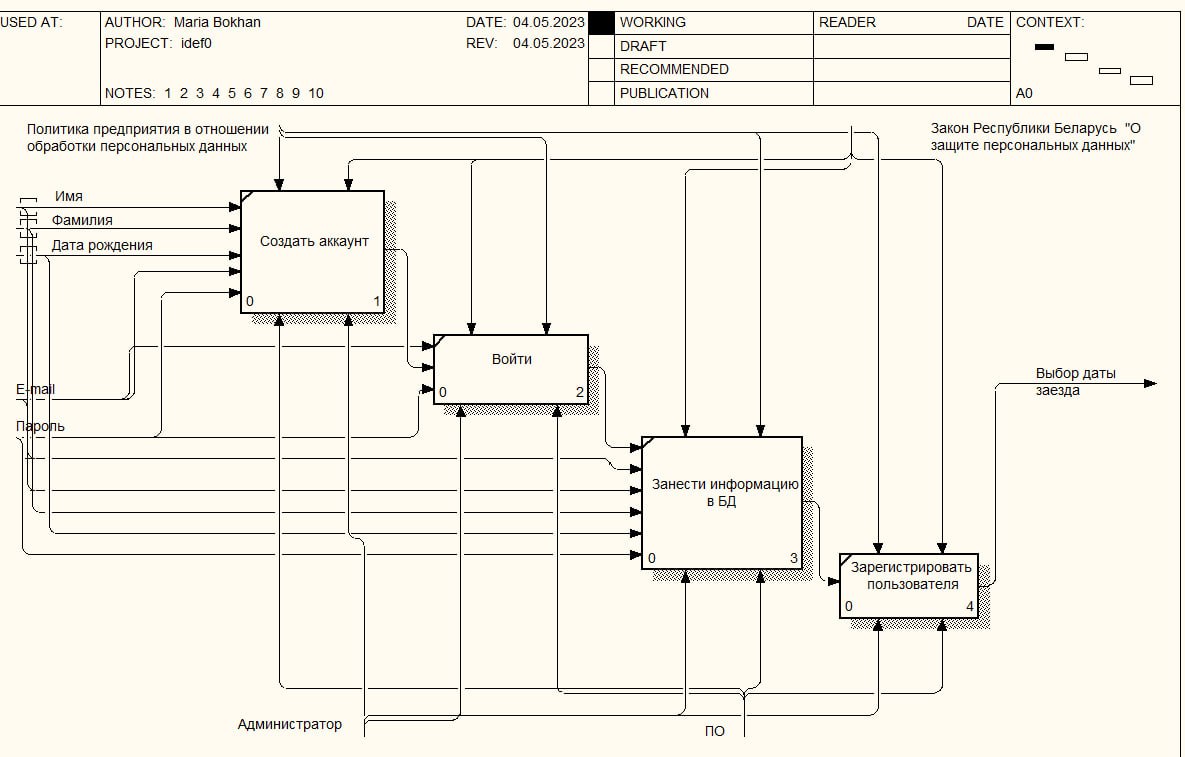
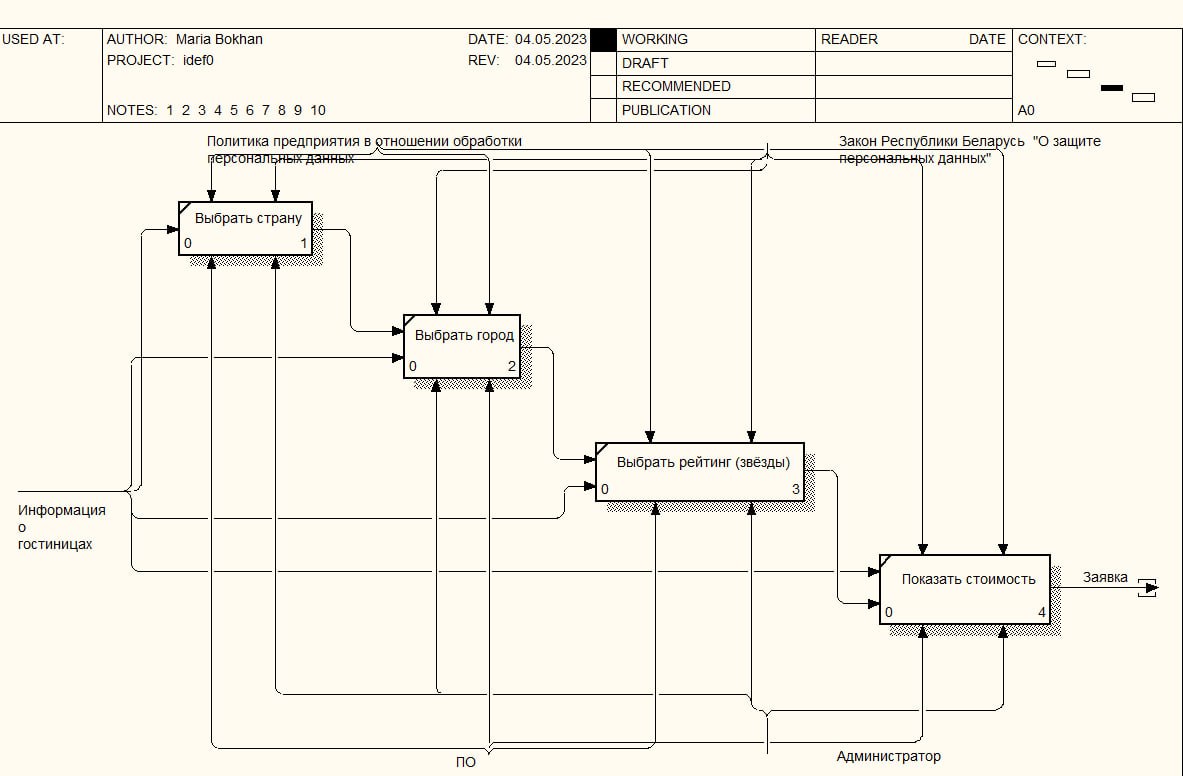
Для подробного представления внутренних процессов системы сделаем декомпозицию блоков 1 (рисунок 2.4) и 3 (рисунок 2.5).

Рисунок 2.4 - Декомпозиция блока «Войти в систему»

Рисунок 2.5 - Декомпозиция блока «Показать гостиницы»

Будем считать, что уровень декомпозиции рассмотренных диаграмм достаточен для отражения цели моделирования, и на диаграммах нижнего уровня в качестве наименований работ используются элементарные функции (с точки зрения пользователя системы).

Главным преимуществом строения диаграмм с помощью программного обеспечения — это возможность проверки правильности ее построения, проверка логического расположения блоков и так далее. Данная функция позволит избежать возможных ошибок в описании бизнес-процесса, сделает его более функциональным и эффективным.

|  |  |
| --- | --- |
| **3** | **РАЗРАБОТКА ИНФОРМАЦИОННОЙ МОДЕЛИ СИСТЕМЫ БРОНИРОВАНИЯ НОМЕРОВ В ГОСТИНИЦЕ** |

Информационная модель описывает структуру и свойства объекта, его функции, процессы и взаимодействие с другими объектами. Информационная модель может быть представлена в виде диаграмм, таблиц, графов и других форматов. Она используется для проектирования систем, разработки программного обеспечения, управления бизнес-процессами и других задач. В случае системы бронирования номеров в гостинице информационная модель помогает определить необходимые функции, связи между ними и требования к базе данных.

Модель базы данных - это абстрактное представление базы данных, которое определяет ее структуру, типы данных, отношения между таблицами и другие характеристики. Она используется для проектирования и создания баз данных, а также для управления данными в них. Модель базы данных может быть представлена в виде диаграмм, таблиц или других форматов. Она позволяет определить связи между таблицами, типы данных, ограничения и правила целостности данных, а также определить доступ к данным и управление ими. В случае системы бронирования номеров в гостинице модель базы данных может помочь определить таблицы для хранения информации о номерах, клиентах, бронированиях и других сущностях, а также связи между ними.

Логическая модель базы данных описывает структуру данных и их отношения на более высоком уровне абстракции, чем физическая модель. Она определяет сущности, атрибуты и связи между ними, а также правила целостности данных. Логическая модель не зависит от конкретной СУБД и может быть использована для проектирования баз данных различных типов.

В качестве основной базы данных была использована программа Postgres SQL. В первую очередь Postgres SQL является одной из самых надежных и стабильных СУБД, что позволяет избежать потери данных и сбоев в работе системы, может легко масштабироваться для работы с большими объемами данных и высокой нагрузкой, поддерживает множество расширений и плагинов, что позволяет адаптировать базу данных под конкретные потребности и требования проекта, обладает высоким уровнем безопасности и поддерживает множество механизмов защиты данных, таких как шифрование, аутентификация и авторизация.

Postgres SQL является открытым и бесплатным программным обеспечением, что позволяет сократить затраты на разработку и поддержку системы.

Для представления структуры хранимых данных программного средства выбрана информационная методология IDEF1X.

IDEF1X (Integrated Definition Language for Information Modeling Express) - язык моделирования данных, разработанный в 1970-х годах для описания структур баз данных. Он используется для создания диаграмм, которые показывают сущности (таблицы), их атрибуты (столбцы) и связи между ними. IDEF1X также предоставляет нотацию для определения ограничений целостности данных, таких как первичные ключи, внешние ключи и уникальные ограничения. Этот язык широко используется в инженерии баз данных и представляет собой стандарт ANSI/ISO.

1. Сущность (Entity) - это объект, который хранится в базе данных и имеет свои атрибуты. Сущность может быть физическим или концептуальным объектом.

2. Атрибут (Attribute) - это свойство сущности, которое описывает ее. Атрибуты могут быть простыми (например, имя, адрес) или составными (например, дата рождения).

3. Связь (Relationship) - это связь между двумя или более сущностями. Она может быть однонаправленной или двунаправленной, и может иметь определенный тип (например, один-ко-многим или многие-ко-многим).

4. Ключ (Key) - это уникальный идентификатор сущности в базе данных. Он может быть первичным ключом (Primary Key), который уникально идентифицирует каждую запись в таблице, или внешним ключом (Foreign Key), который связывает две таблицы между собой.

5. Ограничение целостности (Integrity Constraint) - это правило, которое определяет допустимые значения для атрибутов и связей в базе данных. Ограничения целостности могут быть первичными ключами, внешними ключами, уникальными ограничениями или ограничениями на значения атрибутов.

Ниже на рисунке 3.1 приведена информационная модель базы данных.

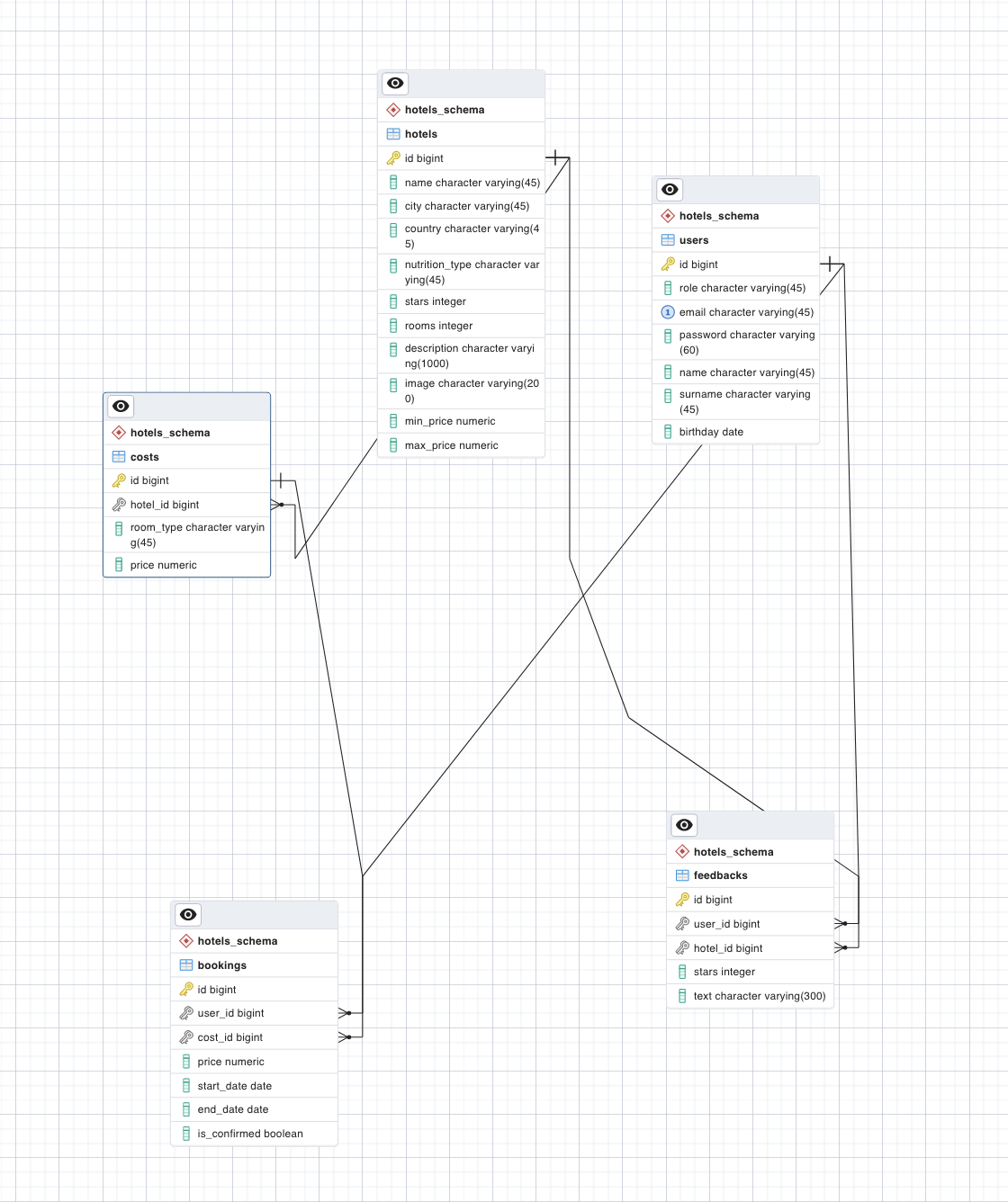


Рисунок 3.1 - Информационная модель базы данных

Перейдём к рассмотрению сущностей, используемых в курсовом проекте.

Сущность «hotels» хранит всю информацию о представленных гостиницах, с которыми пользователь может ознакомиться на сайте. Атрибуты сущности «hotels»:

* id (тип данных bigint) – первичный ключ и уникальный идентификатор гостиницы;
* name (тип данных character varying) – название гостиницы;
* city (тип данных character varying) – город нахождения гостиницы.
* country (тип данных character varying) – страна нахождения гостиницы.
* nutrition\_type (тип данных character varying) – организация питания в гостинице;
* stars (тип данных integer) – рейтинг гостиниц, представленный на сайте;
* rooms (тип данных integer) – категории номеров в гостинице;
* description (тип данных character varying) – описание гостиницы;
* image (тип данных character varying) – фотографии гостиницы;
* min\_price (тип данных numeric) – минимальная цена проживания;
* max\_price (тип данных numeric) – максимальная цена проживания;

Сущность «users» хранит информацию об учетных записях авторизовавшихся пользователей. Атрибуты сущности «users»:

* id (тип данных bigint) – первичный ключ и уникальный идентификатор пользователя;
* role (тип данных character varying) – роль пользователя;
* email (тип данных character varying) – электронный почтовый ящик пользователя;
* password (тип данных character varying) – пароль пользователя;
* name (тип данных character varying) – имя пользователя;
* surname (тип данных character varying) – фамилия пользователя;
* birthday (тип данных date) – дата рождения пользователя;

Сущность «costs» хранит информацию о тарифах на проживание для разных комнат и гостиниц. Атрибуты сущности «costs»:

* id (тип данных bigint) – первичный ключ и уникальный идентификатор стоимости;
* hotels\_id (тип данных bigint) – уникальный идентификатор гостиницы;
* rooms\_type (тип данных character varying) – тип комнаты;
* price (тип данных numeric) – цена;

Сущность «booking» хранит информацию о поступивших бронях и их состояниях. Атрибуты сущности «bookings»:

* id (тип данных bigint) – первичный ключ и уникальный идентификатор брони;
* user\_id (тип данных bigint) – уникальный идентификатор пользователя;
* cost\_id (тип данных bigint) – уникальный идентификатор стоимости;
* price (тип данных numeric) – цена;
* start\_date (тип данных date) – дата заезда;
* end\_date (тип данных date) – дата выезда;
* is\_confirmed (тип данных boolean) – подтверждение брони;

Сущность «feedbacks» хранит отзывы, оставленные проживающими. Атрибуты сущности «feedbacks»:

* id (тип данных bigint) – первичный ключ и уникальный идентификатор брони;
* user\_id (тип данных bigint) – уникальный идентификатор пользователя;
* hotels\_id (тип данных bigint) – уникальный идентификатор гостиницы;
* stars (тип данных integer) – рейтинг, выставляемый пользователем;
* text (тип данных character varying) – сообщение пользователя о гостинице (отзыв);

Ниже определим форму базы данных.

Первая нормальная форма. Устранение повторяющиеся группы в отдельных таблицах. Создание отдельной таблицы для каждого набора связанных данных. Идентификация каждого набор связанных данных с помощью первичного ключа.

Вторая нормальная форма. Создание отдельных таблиц для наборов значений, относящихся к нескольким записям. Связь этих таблиц с помощью внешнего ключа. Записи могут зависеть только от первичного ключа таблицы.

Третья нормальная форма. Устранение полей, не зависящих от ключа. Чтобы нормализовать базу данных до третьей нормальной формы, необходимо сделать так, чтобы в таблицах отсутствовали неключевые столбцы, которые зависят от других неключевых столбцов.

Неключевые столбцы не должны пытаться играть роль ключа в таблице, т.е. они действительно должны быть неключевыми столбцами, такие столбцы не дают возможности получить данные из других столбцов, они дают возможность посмотреть на информацию, которая в них содержится, так как в этом их назначение.

Созданная модель является третьей нормальной формой.

Нормализация не является обязательной, но приносит следующие преимущества:

1. ****Упрощается процесс выборки**.** Речь идет об упрощении работы по составлению запросов, то есть пользователь сможет получать нужную информацию относительно простыми запросами;
2. О****беспечивается целостность данных****. Можно говорить о минимизации искажения информации и снижении вероятности потери данных;
3. ****Улучшается масштабируемость**.** При соблюдении правил нормализации формируются благоприятные предпосылки к росту БД;
4. О****тсутствует избыточность**** (data redundancy). Избыточность —проблема непродуктивного использования свободного места на жестком диске, затрудняющая обслуживание БД.
5. ****Отсутствие несогласованных зависимостей****. Несогласованные зависимости затрудняют доступ к данным, ведь путь к такой информации может быть неправилен и нелогичен.

|  |  |
| --- | --- |
| **4** | **ПРОЕКТИРОВАНИЕ СИСТЕМЫ БРОНИРОВАНИЯ НОМЕРОВ В ГОСТИНИЦЕ НА ОСНОВЕ UML** |

UML (Unified Modeling Language) - это стандартный язык моделирования, используемый для проектирования систем и программного обеспечения. Он позволяет описывать структуру, поведение и взаимодействие компонентов системы.

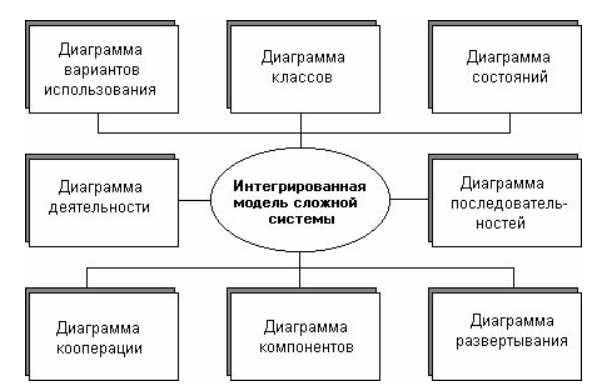
Диаграмма в UML – это графическое представление набора элементов, изображаемое в виде связанного графа с вершинами(сущностями) и ребрами (отношениями), используемое для визуализации системы с разных точек зрения. В UML выделяют 8 типов диаграмм (рисунок 4.1).

Рисунок 4.1 Интегрированная модель сложной системы в нотации UML.

Диаграмма вариантов использования показывает различные сценарии, которые могут произойти при бронировании номера клиентом.

Диаграмма классов показывает классы объектов, которые используются в процессе бронирования. Диаграмма классов также может показать связи между классами, например, что каждое бронирование связано с одним номером и одним клиентом. В целом, диаграмма классов помогает описать структуру системы бронирования номера в гостинице и идентифицировать все необходимые классы объектов для ее реализации.

Диаграмма состояний показывает все возможные состояния, которые может пройти процесс бронирования, начиная от начала процесса и заканчивая получением подтверждения бронирования.

Диаграмма последовательности показывает последовательность действий при бронировании номера клиентом.

Диаграмма развертывания показывает, какие компоненты системы используются для выполнения процесса бронирования и как они развертываются на физических устройствах. Например, диаграмма может показать, что клиент использует веб-интерфейс для ввода информации о бронировании, а сервер приложений обрабатывает эту информацию и сохраняет ее в базе данных. Диаграмма развертывания также может показать, на каких физических устройствах работают компоненты системы, например, веб-сервер может быть развернут на одном сервере, а база данных - на другом. Диаграмма развертывания помогает описать архитектуру системы и идентифицировать возможные проблемы с производительностью или масштабируемостью.

UML позволяет более наглядно и понятно описывать систему и ее компоненты, а также упрощает коммуникацию между разработчиками и заказчиками.

**Диаграмма вариантов использования (use case diagram).**

Для информационной системы интернет-бронирования номеров гостиницы предусмотрены следующие роли (сущности-актеры): клиент, администратор гостиницы. Каждый актер системы (или вариант использования) включает в себя определенный набор действий (рисунок 4.4).

Клиент информационной системы интернет-бронирования номеров гостиницы имеет следующие функции:

* регистрация;
* авторизация;
* просмотр всех;
* подбор по критериям;
* бронь;
* отзыв.

Администратор гостиницы выполняет четыре действия:

* просмотр заявок;
* подтверждение заявки;
* отклонение заявки;
* возможность редактировать информацию о гостинице.

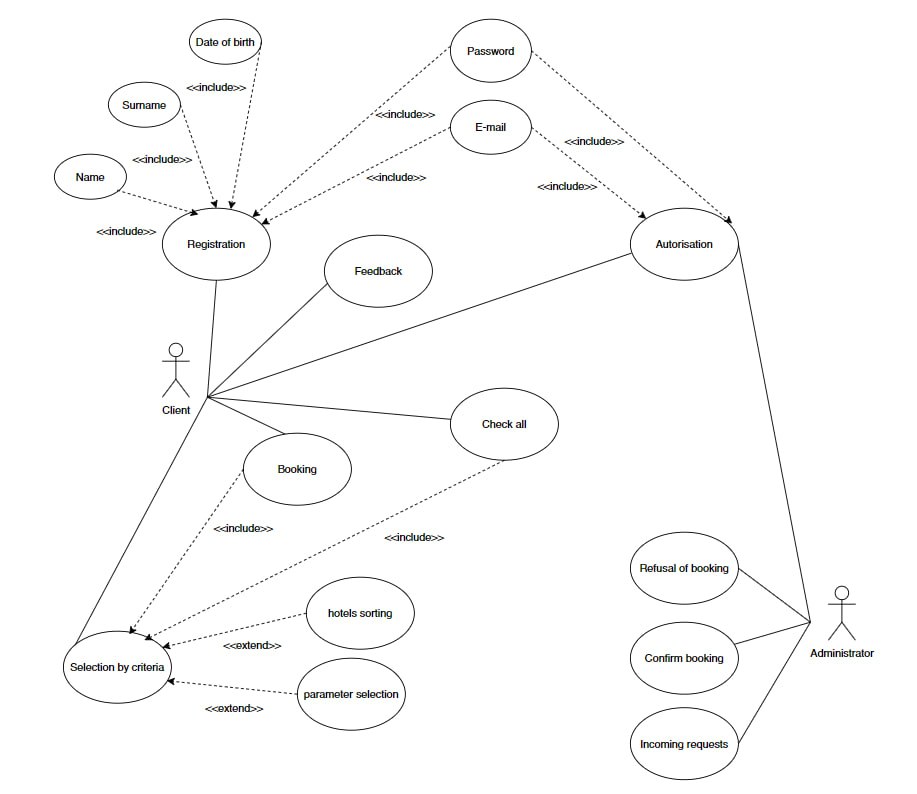


Рисунок 4.5 - Диаграмма вариантов использования

**Диаграмма классов**

Диаграмма классов – структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей между ними. Широко применяется не только для документирования и визуализации, но также для конструирования посредством прямого или обратного проектирования. Целью создания диаграммы классов является графическое представление статической структуры декларативных элементов системы (классов, типов и т. п.). Она содержит в себе также некоторые элементы поведения (например – операции), однако их динамика должна быть отражена на диаграммах других видов (диаграммах коммуникации, диаграммах состояний). При представлении сущностей реального мира разработчику требуется отразить их текущее состояние, их поведение и их взаимные отношения.

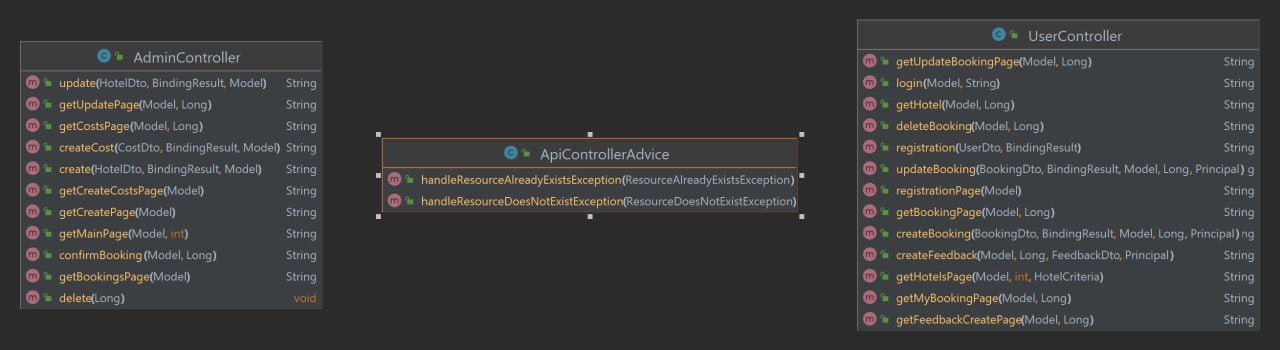
Диаграмма классов серверного программного средства представлена на рисунке

Рисунок 4.2 - Диаграмма классов клиентского приложения

Контроллер смотрит на объект Request, создает и возвращает объект Response. Ответ может быть HTML-страницей, JSON, XML, сохраняемым файлом, редиректом, ошибкой 404 или чем-то другим. Контроллер может запускать любую произвольную логику, которая нужна приложению для отображения содержимого страницы.

Класс AdminController предназначен для управления работы открывающихся окон со стороны администратора.

Класс UserController предназначены для управления работой каждого окна, представленного для клиента.

Класс ApiControllerAdvice представляет собой класс рекомендаций контроллера для обработки исключений. Он перехватывает определенные исключения, создаваемые приложением, и сопоставляет их с соответствующим кодом состояния HTTP и сообщением. Сообщения, возвращаемые этими обработчиками, используются для информирования клиента об ошибке.

Сущности – это абстракции, которые являются основными элементами модели, связи соединяют их между собой, а диаграммы группируют представляющие интерес наборы сущностей. Диаграмма сущностей является интерпритацией таблиц из базы данных (рисунок ).

БД позволяет хранить информацию в таблицах, для того чтобы это осуществить, у hibernate существует специальный механизм — mapping. Mapping-проецирование (сопоставление) Java классов с таблицами базы данных. Реализация маппинга произведена через аннотации: @Entity, @Table.

Рисунок 4.3 - Диаграмма классов сущностей

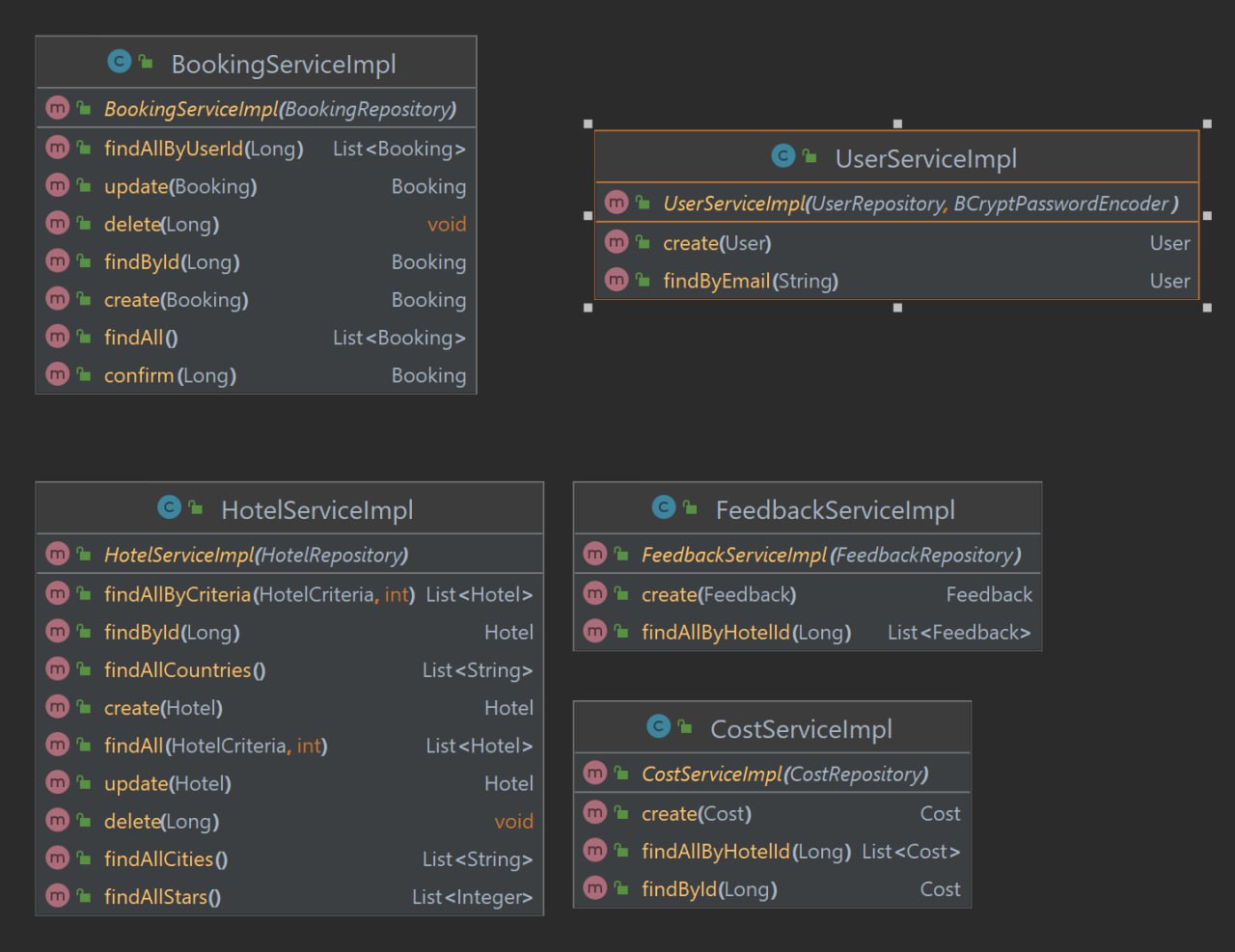
В полях классов сервисного уровня расположены объекты получения данных, которые содержат в себе все CRUD операции. Сервисный уровень объединяет все эти методы, чтобы определить более сложный функционал системы.

Рисунок 4.4 - Диаграмма классов сервисного уровня

**Диаграмма развёртывания**

1. Показывает физическую архитектуру системы и ее развертывание на серверах и устройствах.
2. Компоненты системы: сервер базы данных, сервер приложений, клиентское приложение.
3. Связи между компонентами: клиентское приложение обращается к серверу приложений, который в свою очередь обращается к серверу базы данных.
4. На диаграмме также показано, какие компоненты развертываются на каких устройствах (например, клиентское приложение может быть установлено на мобильном устройстве или на компьютере).

На рисунке 4.6 представлена диаграмма развертывания, где в качестве девайса выступает машинное обеспечение пользователя. В качестве сервера была использована программа Docker, которая позволяет облегчить и ускорить работу приложения. Данные, которые отправляются серверу, потом поступают в базу данных. Базой данных в данной курсовой работе выступает PostgreSQL.

Сервер взаимодействует с устройством посредством протокола HTTP.



Рисунок 4.6 - Диаграмма развертывания

**Диаграмма состояний**

Диаграмма состояний (Statechart diagram) – это диаграмма, предназначенная для моделирования динамических аспектов поведения систем. Она позволяет описать все возможные состояния объекта или системы и переходы между ними. Диаграмма состояний в UML помогает описать поведение объекта, его жизненный цикл и действия, которые должны быть выполнены при переходе между состояниями.

Основными элементами диаграммы состояний являются:

1. Состояние (State) - это условие, в котором находится объект или система. Состояние может быть описано в текстовом виде или графически представлено в виде прямоугольника с названием состояния.
2. Переход (Transition) - это действие, которое вызывает изменение состояния объекта или системы. Переход может быть вызван внешними событиями или внутренними условиями и описывается стрелкой, которая указывает направление перехода между состояниями.
3. Событие (Event) - это внешнее воздействие, которое вызывает переход между состояниями. Событие может быть вызвано пользователем, другими объектами или системой.
4. Действие (Action) - это действие, которое должно быть выполнено при переходе между состояниями. Действие может быть выполнено внутри объекта или системы, и описывается в виде текста или графически.

Данная диаграмма описывает состояние бронирования гостиничного номера. Сначала пользователь выбирает дату своего заезда и выезда, если данные были введены правильно, то он переходит на следующий этап. Пользователь дальше подбирает гостиницу по критериям, сервер их обрабатывает, в случае успеха пользователю выводятся подходящие для него варианты. Если же гостиница, подходящая по всем критериям, не будет найдена, то пользователь возвращается на первоначальный этап. Далее пользователь выбирает необходимую для него гостиницу и оставляет заявку на бронирование. Затем сервером обрабатываются полученные данные. После решения администратора, на сервере подтверждается бронирование, и пользователь получает его. Если же администратор отказал в брони, то пользователь должен повторно ввести свои данные. Далее на сервере просматривается статус бронирования пользователя.

Рисунок 4.6 - Диаграмма состояния процесса бронирования

**Диаграмма последовательности**

Диаграмма последовательности представляет собой конструкцию UML.

Диаграммы последовательностей, обычно используемые разработчиками, моделируют взаимодействия между объектами в едином сценарии использования. Они иллюстрируют, как различные части системы взаимодействуют друг с другом для выполнения функции, а также порядок, в котором происходит взаимодействие при выполнении конкретного случая использования.

Для диаграммы последовательности ключевым моментом является именно динамика взаимодействия объектов во времени. При этом диаграмма последовательности имеет два направления. Одно – слева направо в виде вертикальных линий, каждая из которых изображает линию жизни отдельного объекта, участвующего во взаимодействии. Графически каждый объект изображается прямоугольником и располагается в верхней части своей линии жизни.

Пользователь взаимодействует с системой бронирования. Чтобы забронировать номер в гостинице, ему надо войти в систему, затем выбрать дату проживания, критерии гостиницы. После он может оставить свою заявку на бронирование.

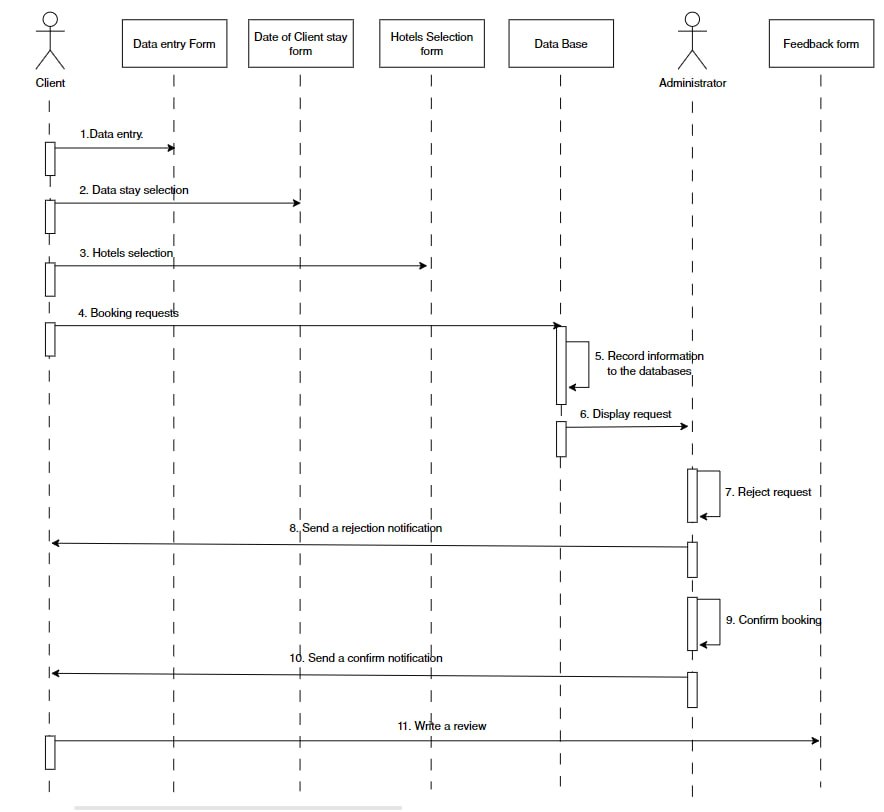


Рисунок 4.7– Диаграмма последовательности процесса бронирования номера в гостинице

В данном главе UML-диаграммы позволяют более наглядно и понятно описывать систему и ее компоненты, а также упрощает коммуникацию между разработчиками и заказчиками.

|  |  |
| --- | --- |
| **5** | **ОБОСНОВАНИЕ ВЫБОРА КОМПОНЕНТТОВ И ТЕХНОЛОГИЙ ДЛЯ РЕАЛИЗАЦИИ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ БРОНИРОВАНИЯ ГОСТИНИЧНЫХ НОМЕРОВ** |

Разрабатываемая в курсовом проекте автоматизированная система должна осуществлять бронь гостиничных номеров.

Согласно требованию на курсовой проект, серверная часть автоматизированной системы должна быть построена на основе набора спецификаций Java EE или фреймворка Spring.

В проекте были выбраны инструменты, предоставляемые фреймворком Spring, так как он широко используется для создания enterprise-приложений.

Spring — это популярный фреймворк для разработки на Java, который используют для создания корпоративных приложений, например CRM. П рименять его могут не только Java-разработчики, но и те, кто работает с Kotlin или Groovy.

Spring Boot - это технология, которая позволяет упростить и ускорить процесс разработки приложений на Java. Она основана на Spring Framework и предоставляет инструменты для автоматической конфигурации и управления зависимостями. Сам Spring Boot представляет собой набор утилит, автоматизирующих настройки фреймворка. Вот что он берёт на себя:

* упаковывает зависимости в starter-пакеты;
* автоматически конфигурирует приложения с помощью jar-зависимостей;
* создаёт веб-сервер, что позволяет локально запускать на нём приложения.

Встроенный в Spring Boot сервер Apache Tomcat работает в качестве веб-сервера и прослушивает запросы на порту localhost 8080.

Для реализации автоматизированной системы бронирования гостиничных номеров на базе Spring Boot можно использовать следующие технологии:

Spring MVC - это фреймворк для разработки веб-приложений на Java. Он позволяет быстро создавать контроллеры, сервлеты, представления и работать с данными.

Spring Data JPA - это инструмент для работы с данными на Java. Он позволяет работать с базами данных через объектно-реляционное отображение (ORM) и генерировать SQL-запросы автоматически.

Thymeleaf - это движок шаблонов для веб-приложений на Java. Он позволяет создавать динамические страницы, включая данные из модели и контроллера.

В качестве основной базы данных была использована программа Postgres SQL. В первую очередь Postgres SQL является одной из самых надежных и стабильных СУБД, что позволяет избежать потери данных и сбоев в работе системы, может легко масштабироваться для работы с большими объемами данных и высокой нагрузкой, поддерживает множество расширений и плагинов, что позволяет адаптировать базу данных под конкретные потребности и требования проекта, обладает высоким уровнем безопасности и поддерживает множество механизмов защиты данных, таких как шифрование, аутентификация и авторизация.

Postgres SQL является открытым и бесплатным программным обеспечением, что позволяет сократить затраты на разработку и поддержку системы.

|  |  |
| --- | --- |
| **6** | **ОПИСАНИЕ АЛГОРИТМОВ, РЕАЛИЗУЮЩИХ, РЕАЛИЗУЮЩИХ КЛЮЧЕВУЮ БИЗНЕС-ЛОГИКУ СИСТЕМЫ БРОНИРОВАНИЯ НОМЕРОВ В ГОСТИНИЦЕ** |

|  |  |
| --- | --- |
| **7** | **РУКОВОДСТВО ПО РАЗВЁРТЫВАНИЮ СИСТЕМЫ БРОНИРОВАНИЯ АВИАБИЛЕТОВ** |

**Руководство по развертыванию программного средства**

С помощью Docker, программного обеспечения для автоматизации развертывания, можно развернуть приложение на любом устройстве, где установлен Docker. Для полного развертывания приложения на сервере нужно выполнить команду «docker compose up» в консоли. Docker установит все зависимости и создаст базу данных, необходимую для приложения. После этого приложение будет готово к использованию.

Этот руководство поможет развернуть автоматизированную систему бронирования в Docker.

1. Установите Docker на вашей машине, используя инструкции на сайте Docker.
2. Скачайте и распакуйте файлы системы бронирования из источника. Система бронирования должна содержать все необходимые файлы, включая Dockerfile.
3. Откройте терминал и перейдите в папку с файлами системы бронирования.
4. Создайте образ Docker используя команду: docker build -t booking\_system.
5. Запустите контейнер на основе образа командой: docker run -p 8080:80 booking\_system. Эта команда запускает контейнер и привязывает его к порту 8080 на локальной машине. Вы можете изменить порт на свой выбор.
6. Перейдите по адресу http://localhost:8080 в вашем браузере для того, чтобы убедиться, что система бронирования работает нормально.
7. Если вы хотите остановить контейнер, выполните команду: docker stop <container\_id>, где <container\_id> – это идентификатор контейнера.

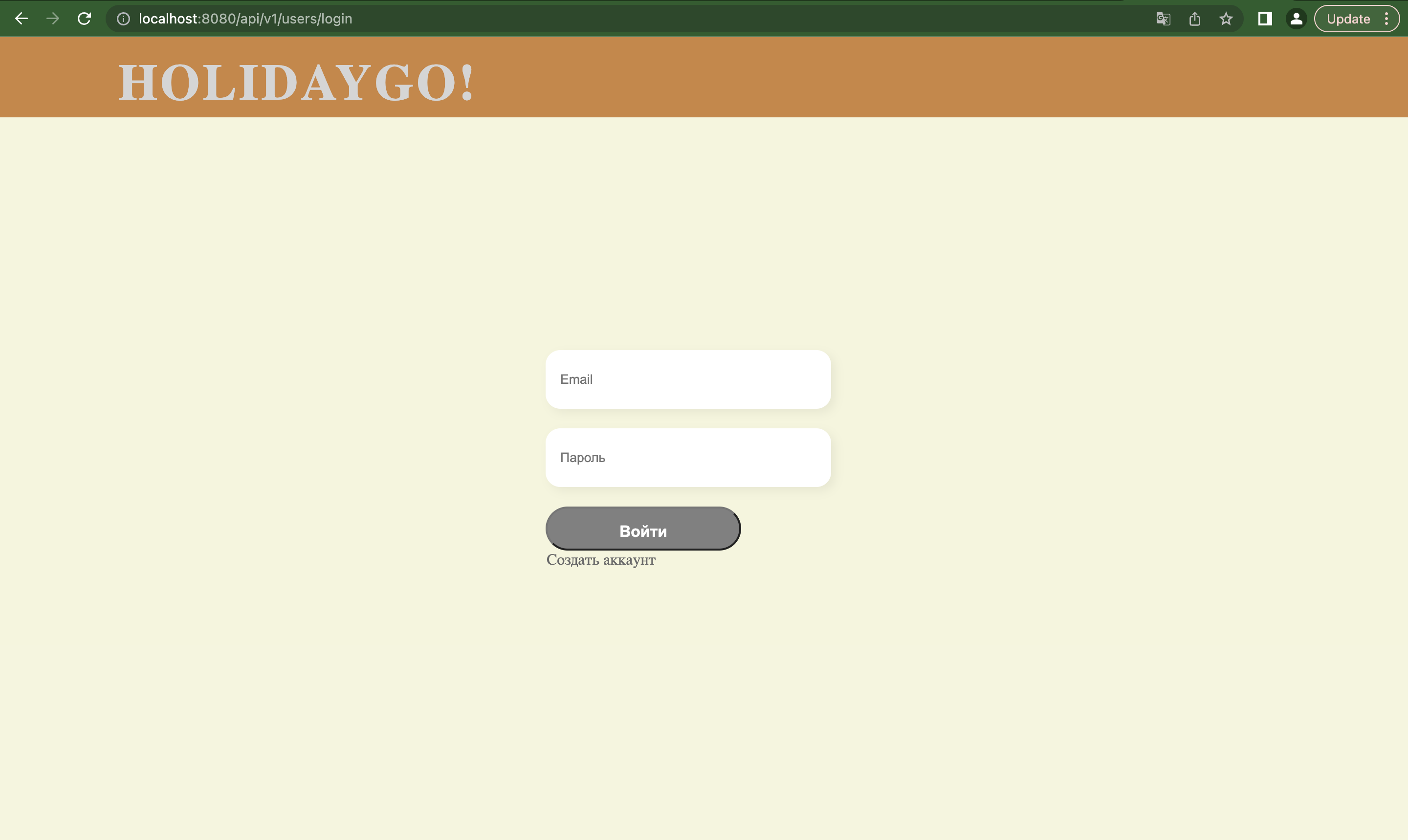
Теперь вы можете использовать автоматизированную систему бронирования в Docker. Этот Dockerfile настроен на использование JDK 17 и Alpine Linux. Он также настроен на использование порта 8080.

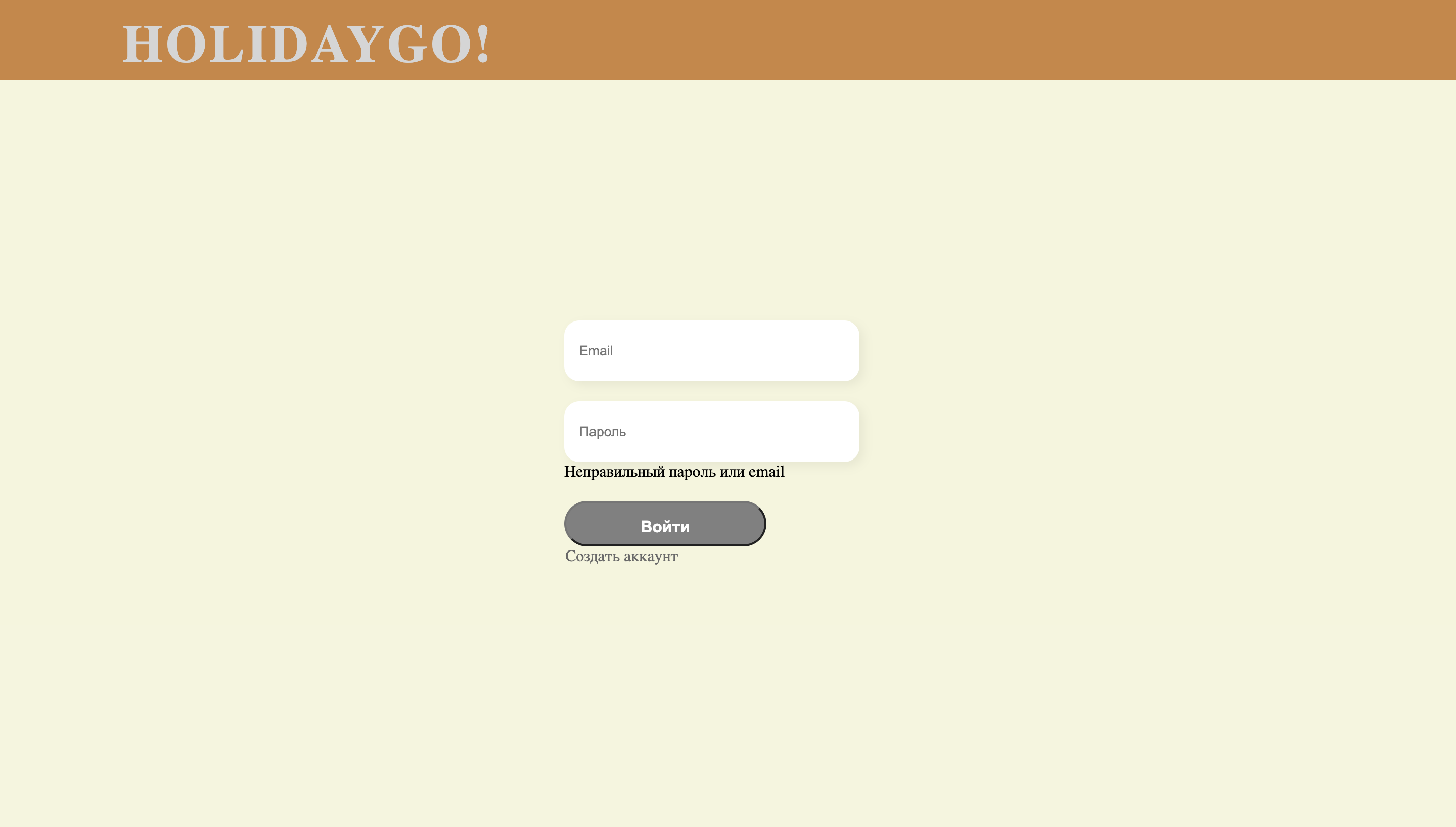
Данный программный продукт использует PostgreSQL для работы с базой данных. Также необходимо наличие на компьютере для клиента и сервера JDK версии 17 и выше.

|  |  |
| --- | --- |
| **8** | **РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ СИСТЕМЫ БРОНИРОВАНИЯ АВИАБИЛЕТОВ** |

**Роль «Клиент»**

После выполнения всех этапов развертывания программы перед клиентом приложения появляется страница автторизации пользователя (рисунок 8.1), которая запрашивает электронную почту и пароль, а также дает переход в другую форму по кнопке «Создать аккаунт» для неавторизованных пользователей. Для избегания системных ошибок выполняется проверка на корректность введения электронного почтового адреса (Правильная валидация email: @*)*. Система выдает ошибку, если пользователь введет адрес неправильно. (рисунок 8.2)

Рисунок 8.1 - Форма входа в систему

Рисунок 8.2 – Ошибка авторизации

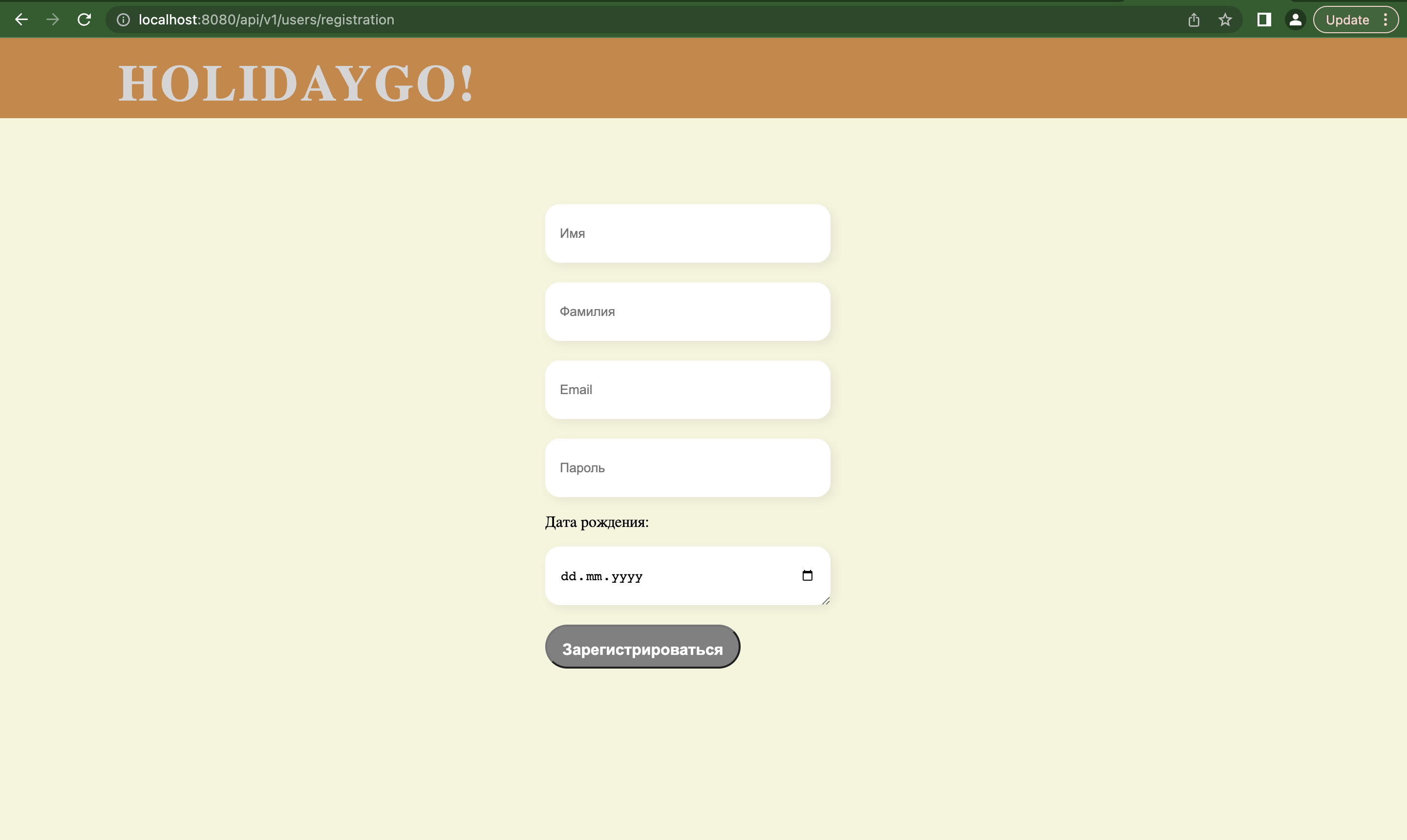
На рисунке 8.3 показана форма регистрации. Новый пользователь должен ввести имя, фамилию, электронный почтовый адрес, пароль и дату своего рождения.

Рисунок 8.3 – Форма регистрации нового пользователя

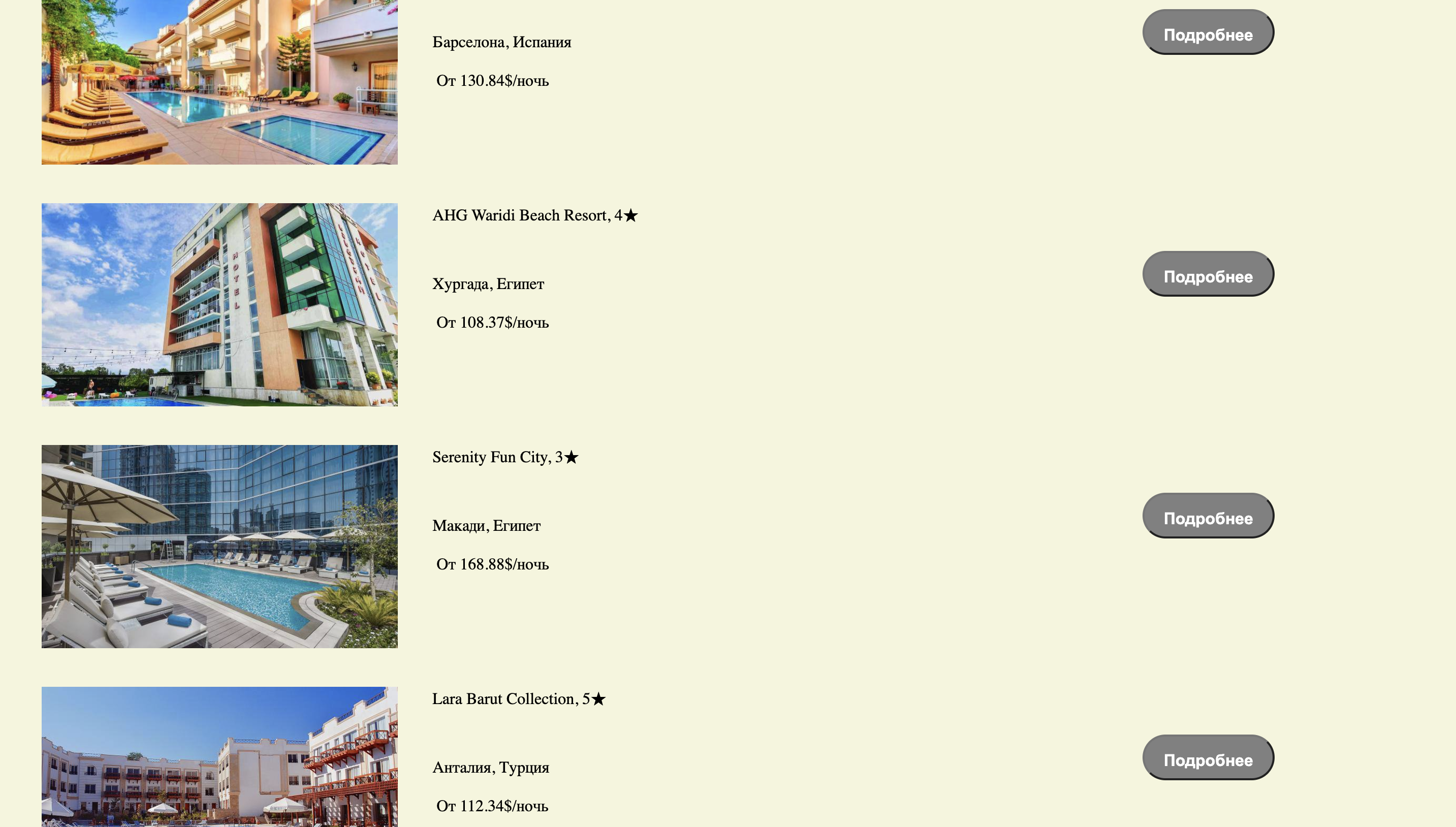
После успешного входа в систему клиенту становится доступным просмотр его бронирований. Также ниже представлен фильтр для выбора гостиницы по критериям: страна, город, рейтинг, дата заезда/дата выезда. При прокрутке страницы можно обнаружить список всех представленных гостиниц, где есть фотографии и указаны минимальный тариф на проживание, рейтинг, страна и город.

Рисунок 8.4 – Список всех доступных гостиниц

Также доступен просмотр каждой гостиницы в отдельности (рисунок 8.5), где можно узнать больше информации (включено ли питание, какие достопримечательности есть рядом).

Рисунок 8.5 – Информация о гостинице на отдельной странице

После выбора гостиницы пользователь должен перейти в форму бронирования (рисунок 8.6). Здесь пользователь может выбрать категорию номера в соответствии со стоимостью, определиться с датой заезда (выбрать дату нужно корректно, чтобы дата заезда была раньше, чем дата выезда). После правильно введённых данных запрос клиента отправляется администратору.

Рисунок 8.6 – Форма бронирования номера в гостинице

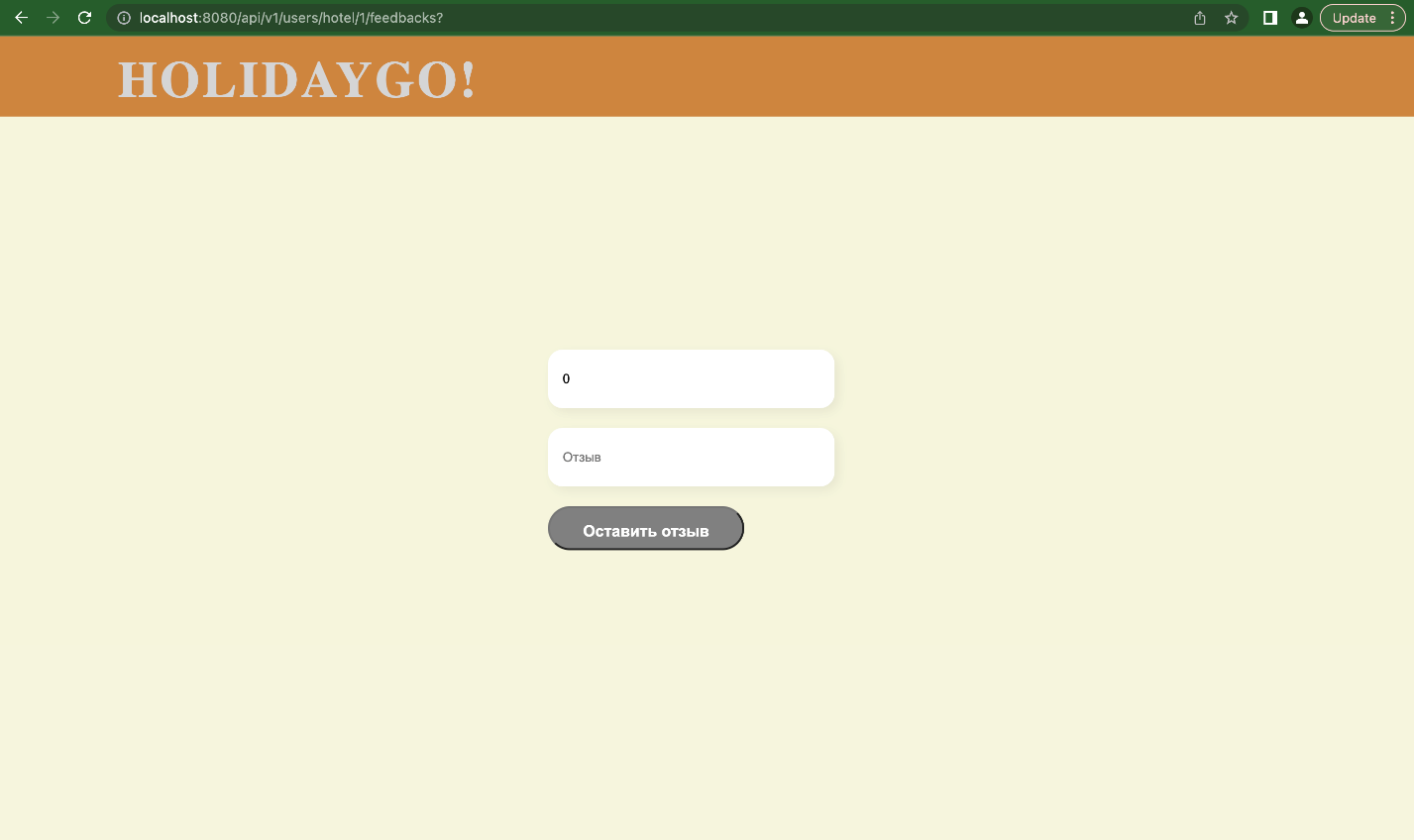
После подтверждения брони со стороны администратора, пользователю открывается возможность оставить свой текстовый отзыв о гостинице и выставить рейтинг.

Рисунок 8.7 – Форма для отзывов.

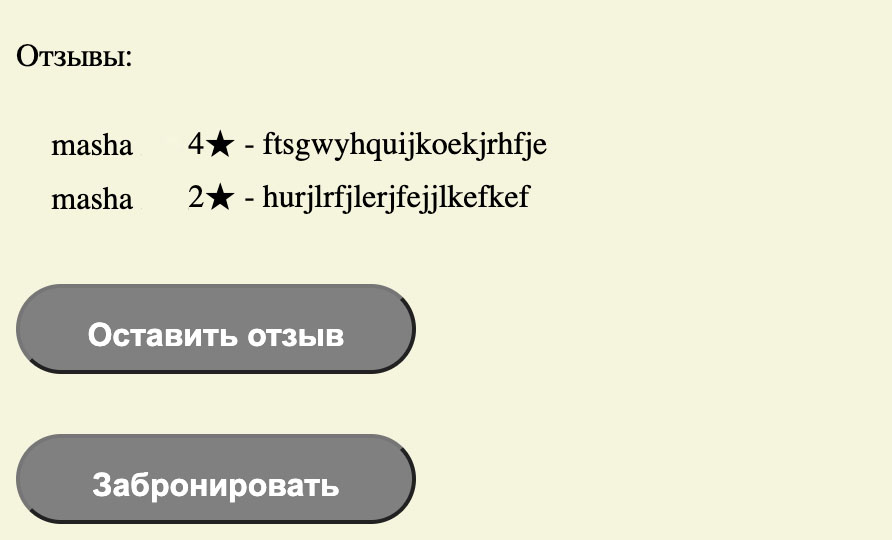
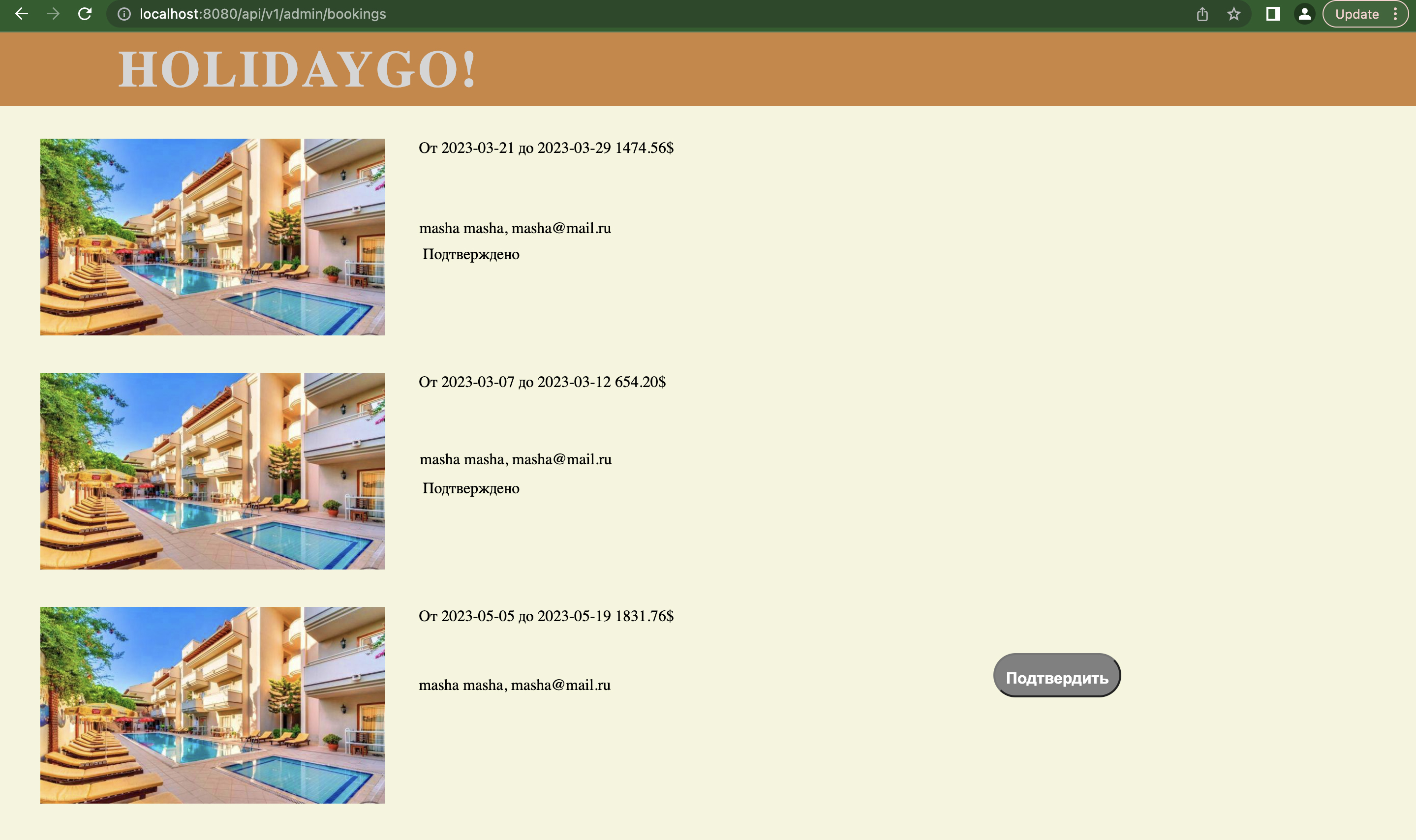
Гость также может просматривать отзывы, оставленные другими пользователями, также эта форма предлагает вновь забронировать номер в выбранной им гостинице (рисунок 8.8).

Рисунок 8.8 – Форма просмотра отзывов

**Роль «Администратор»**

Пользователи с ролью «Администратора» имеют те же функции, что и другие пользователи. Однако, администратор имеет возможность управления входящими заявками на бронирование: администратор может подтвердить бронь (рисунок 8.9).



По результатам работы тестирования программного продукта, видно, что поставленные задачи решены в полном объеме. Разработанная система позволит пользователям регистрироваться и авторизоваться в системе, изменять свои личные данные, бронировать авиабилеты, добавлять новые и анализировать данные. Данное приложение облегчит работу гостиницам и сделает возможность быстрого и легкого бронирования для пользователей

**ПРИЛОЖЕНИЕ А**