МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГУ»)

Факультет компьютерных наук

Кафедра информационных систем

Курсовая работа по дисциплине «Технологии программирования»

Разработка веб-приложения «Сервис поиска отелей Bochka»

09.03.02 Информационные системы и технологии

6 семестр 2023/2024 учебного года

Зав. кафедрой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ к. т. н., доцент Д. Н. Борисов

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ст. 3 курса оч. отд. Порядин А.В.

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ст. 3 курса оч. отд. Исаченко Б.В.

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ст. 3 курса оч. отд. Ткаченко А.Ю.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ст. преподаватель Тарасов В.С.

Воронеж 2024

Введение

В нашем современном мире, где путешествия стали неотъемлемой частью жизни многих людей, поиск подходящего места для проживания во время поездок становится всё более актуальной задачей. Сервисы бронирования отелей и других вариантов размещения играют ключевую роль в планировании путешествий и помогают экономить время и средства. Однако, в условиях растущей конкуренции и разнообразия предложений, выбор подходящего сервиса может стать настоящим испытанием.

Сервис поиска отелей Bochka направлен на решение этой проблемы, предоставляя пользователям удобный и функциональный инструмент для поиска и бронирования отелей по всему миру. Bochka отличается от других сервисов тем, что сочетает в себе широкий выбор предложений, конкурентоспособные цены и простоту в использовании.

**Актуальность** **обусловлена** необходимостью создания эффективного и удобного сервиса для поиска отелей, который помог бы путешественникам экономить время и средства, а также сделал бы процесс планирования поездок более приятным и удобным. Разработка сервиса Bochka отвечает этой потребности, предоставляя пользователям широкий выбор вариантов размещения и удобный инструмент для их поиска и бронирования.

**Целью курсовой работы** **является** разработка функционального сервиса поиска отелей Bochka. В рамках работы будет разработан пользовательский интерфейс, обеспечивающий удобное взаимодействие с сервисом, а также реализован функционал для поиска и бронирования отелей. Сервис Bochka будет способствовать повышению эффективности процесса планирования путешествий и поможет путешественникам найти подходящий вариант размещения в соответствии с их потребностями и бюджетом.

1. Постановка задачи
   1. Цели создания приложения

Целями создания приложения являются:

* Создание системы, которая позволит пользователям легко и удобно бронировать отели для размещения во время путешествий или поездок;
* Увеличение доходов заказчика за счет продажи гостиничных номеров через онлайн-платформу;
* Продажа гостиничных номеров для конкретной аудитории, предпочитающей определенные условия размещения, такие как питание, наличие специальных удобств и т.д., что позволяет удовлетворить разнообразные потребности клиентов;
* Обеспечение информативной карточки каждого отеля с подробным описанием, фотографиями и отзывами, что помогает пользователям принимать осознанные решения.
  1. Задачи приложения

Приложение позволяет решать следующие задачи:

* Искать отели с глубокой фильтрацией по времени заселения, местоположению, наличию техники, парковке и другим параметрам;
* Просматривать подробную информацию о каждом отеле, включая фотографии, описание, услуги и прочее;
* Бронировать отель непосредственно через веб-сайт;
* Оставлять отзывы и оценки о каждом отеле;
* Сохранять историю бронирования для последующего доступа;
* Создавать учётную запись пользователя и осуществлять редактирование её данных, после регистрации в системе;
* Создавать учётную запись владельца отеля для добавления отеля в систему;
* Коммуницировать между пользователем и владельцем отеля в чате.
  1. Требования к приложению
     1. Требования к приложению в целом

Разрабатываемое приложение должно удовлетворять следующим основным требованиям:

* Приложение должно корректно работать в современных веб-браузерах;
* Приложение должно реализовывать основные функциональные задачи, соответствующие целям проекта;
* Созданное приложение должно иметь архитектуру, соответствующую шаблону клиент-серверного приложения, с разделением на back-end и front-end;
* Взаимодействие между back-end и front-end должно осуществляться посредством REST API.
  + 1. Требования к функциям (задачам), выполняемым приложением

Разрабатываемое приложение должно соответствовать следующим функциональным требованиям:

Неавторизованный пользователь должен обладать возможностью:

* Авторизоваться/зарегистрироваться в приложении;
* Получать информацию о предложениях отелей с глубокой фильтрацией по различным критериям, таким как местоположение, цена, удобства и другие параметры;
* Просматривать детальную информацию о каждом отеле, включая фотографии, описания, отзывы и оценки;
* Выполнять поиск отелей по различным критериям.

Авторизованный пользователь (в роли клиента) должен обладать возможностью:

* Получать информацию о предложениях отелей с глубокой фильтрацией по различным критериям, таким как местоположение, цена, удобства и другие параметры;
* Просматривать детальную информацию о каждом отеле, включая фотографии, описания, отзывы и оценки;
* Выполнять поиск отелей по различным критериям;
* Просматривать свою историю бронирования;
* Возможность бронировать отель;
* Редактировать персональную информацию в учётной записи;
* Оставлять отзывы и оценки об отелях;
* Возможность общаться с владельцем отеля через чат.

Авторизованный пользователь (в роли владельца отеля) должен обладать возможностью:

* Добавлять информацию о своем отеле, включая описание, фотографии, цены и доступные удобства;
* Обновлять информацию о своем отеле, в том числе актуализация цен и доступности номеров;
* Просматривать и управлять бронированиями;
* Общаться с клиентами через систему чата.
  + 1. Требования к структуре

Для Frontend:

Сервис должен быть реализован в соответствии с архитектурным паттерном Module — паттерн, который используется для организации кода в отдельные модули или компоненты. Цель использования паттерна Module - избежать конфликтов и обеспечить лучшую структурированность, масштабируемость и повторное использование кода. Каждый модуль содержит свою собственную область видимости, что позволяет исключить конфликты между переменными или функциями из разных модулей.

Для Backend:

Приложение должно быть реализовано в соответствии с подходом MVC (Model – View – Controller) — паттерн разработки, разделяющий архитектуру приложения на три модуля: модель (Model), представление или вид (View), контроллер (Controller).

* Model – это основная логика приложения. Отвечает за данные, методы работы с ними и структуру программы. Модель реагирует на команды из контроллера и выдает информацию и/или изменяет свое состояние. Она передает данные в представление;
* View – отвечает за визуализацию информации, которую он получает от модели. View отображает данные на уровне пользовательского интерфейса. Например, в виде таблицы или списка. Представление определяет внешний вид приложения и способы взаимодействия с ним;
* Controller – обеспечивает взаимодействие с системой: обрабатывает действия пользователя, проверяет полученную информацию и передает ее модели. Контроллер определяет, как приложение будет реагировать на действия пользователя. Также контроллер может отвечать за фильтрацию данных и авторизацию.
  + 1. Требования к программному обеспечению сайта

Для реализации серверной части приложения будут использоваться следующие средства:

* Язык программирования Python;
* Фреймворк FastAPI;
* СУБД PostgreSQL;
* Инструмент для создания документации API Swagger.

Для реализации клиентской части приложения будут использоваться следующие средства:

* Язык программирования JavaScript;
* Фреймворк React.

Для развёртывания приложения будут использоваться следующие средства:

* Клиент Certbot для создания и получения SSL сертификата;
* Docker для автоматизации развёртывания;
* Nginx для обеспечения поддержки SSL и проксирования запросов к back-end приложению.

Инструменты для ведения документации:

* Miro – платформа для совместной работы распределенных команд;
* Swagger – фреймворк для спецификации REST API;
* Figma – онлайн-сервис для дизайнеров, веб-разработчиков и маркетологов. Он предназначен для создания прототипов сайтов или приложений, иллюстраций и векторной графики.
* Дополнительный инструментарий:
* Git – распределённая система управления версиями;
* GitHub – платформа разработки программного обеспечения с открытым исходным кодом, представляющая систему управления репозиториями программного кода для Git;
* GitHub Projects – визуальный инструмент, обеспечивающий эффективность командной работы на любом проекте.

В качестве преимуществ выбранных технологий можно отметить следующее:

Для Python, FastAPI:

* Готовые решения для реализации RESTful архитектуры;
* Скорость написания программного кода.

Для PostgreSQL:

* Функциональность;
* Высокая надежность и производительность;
* Бесплатное и открытое ПО.

Для JavaScript и React:

* Кросс-платформенность;
* Поддержка разными браузерами.
  + 1. Требования к оформлению и верстке страниц

Все страницы сайта должны быть выполнены в едином стиле, соответствующем тематике отельного бронирования. Цветовая палитра и стили шрифтов должны быть гармонично подобраны и привлекательны для пользователей. Приложение должно содержать разработанный логотип, отражающий его назначение и стиль.

Необходимо корректное и одинаковое отображение страниц сайта в

следующих браузерах:

* Google Chrome 122.0.6261.128/129;
* Yandex Browser 23.11.3.955;
* Microsoft Edge 121.0.2277.83;
* Safari 16.5.2;
* Mozilla Firefox 123.0.1.

Верстка сайта должна быть адаптирована под популярные разрешения экранов, чтобы обеспечить удобство использования и приятный внешний вид для всех пользователей.

* + 1. Требования к защите информации

Для обеспечения безопасности информации будет использоваться механизм JWT-токенов. Даже в случае получения злоумышленником такого токена, который предоставляет доступ ко всем функциям приложения, его действие будет ограничено заданным периодом времени, после чего токен станет недействительным и потребуется получить новый.

* 1. Задачи, решаемые в процессе разработки

Были поставлены следующие задачи:

* Анализ предметной области;
* Обзор аналогов;
* Постановка задачи;
* Создание репозитория GitHub и доски в GitHub Projects;
* Разработка требований: к приложению в общем, к функциям, к структуре, к программному обеспечению, к оформлению и верстке страниц, к защите информации;
* Создание диаграмм: прецедентов, состояний, активностей, последовательностей, классов, развертывания;
* Разработка дизайна приложения;
* Написание технического задания в соответствии с ГОСТ 34.602 – 2020;
* Реализация интерфейса приложения;
* Реализация серверной части приложения;
* Развертывание приложения;
* Написание курсовой работы.

1. Анализ предметной области
   1. Глоссарий

В настоящей работе используются следующие термины и сокращения с соответствующими определениями:

* **API** – программный интерфейс приложения. Описание способов, которыми одна компьютерная программа может взаимодействовать с другой программой;
* **Frontend** – это клиентская часть продукта (интерфейс, с которым взаимодействует пользователь);
* **Backend** – программно-аппаратная часть приложения (логика приложения, скрытая от пользователя);
* **Авторизация** – Предоставление определённому лицу прав на выполнение определённых действий; а также процесс проверки (подтверждения) данных прав при попытке выполнения этих действий;
* **Авторизованный пользователь** – Пользователь, который успешно прошел процесс авторизации в приложении, предоставив свои учетные данные и подтвердив свою идентичность. Авторизованный пользователь имеет доступ ко всем основным функциям приложения;
* **Глубокая** **фильтрация** – функциональность в веб-приложениях, которая позволяет пользователям настраивать широкий спектр параметров для поиска и отображения результатов, соответствующих их конкретным потребностям и предпочтениям;
* **Искусственный интеллект** – Набор технологий и алгоритмов, которые позволяют приложению анализировать большие объемы данных, выявлять закономерности и предсказывать поведение пользователей для персонализации рекомендаций и улучшения качества обслуживания;
* **Неавторизованный пользователь** – Пользователь, который еще не прошел процесс авторизации в приложении или не предоставил верные учетные данные для подтверждения своей идентичности. Неавторизованный пользователь имеет ограниченный доступ к функциям приложения;
* **Профиль (в веб-приложении)** – Учетная запись пользователя в веб-приложении, вход в которую осуществляется с помощью логина / номера телефона / e-mail и пароля. В учетной записи содержится информация о пользователе;
* **Cерверная** часть– это программа, которая обеспечивает взаимодействие клиента и сервера;
* **Сервер** – это устройство, в частности компьютер, которое отвечает за предоставление услуг, программ и данных другим клиентам посредством использования сети;
* **Фреймворк** – Программные продукты, которые упрощают создание и поддержку технически сложных или нагруженных проектов. Фреймворк, как правило, содержит только базовые программные модули.
  1. Обзор аналогов

На рынке существует ряд популярных приложений для поиска и бронирования жилья, среди которых выделяются такие аналоги, как Booking.com, Airbnb и Яндекс.Путешествия. Рассмотрим их особенности, достоинства и недостатки для того, чтобы определить направления улучшения приложения "Bochka".

* + 1. Booking

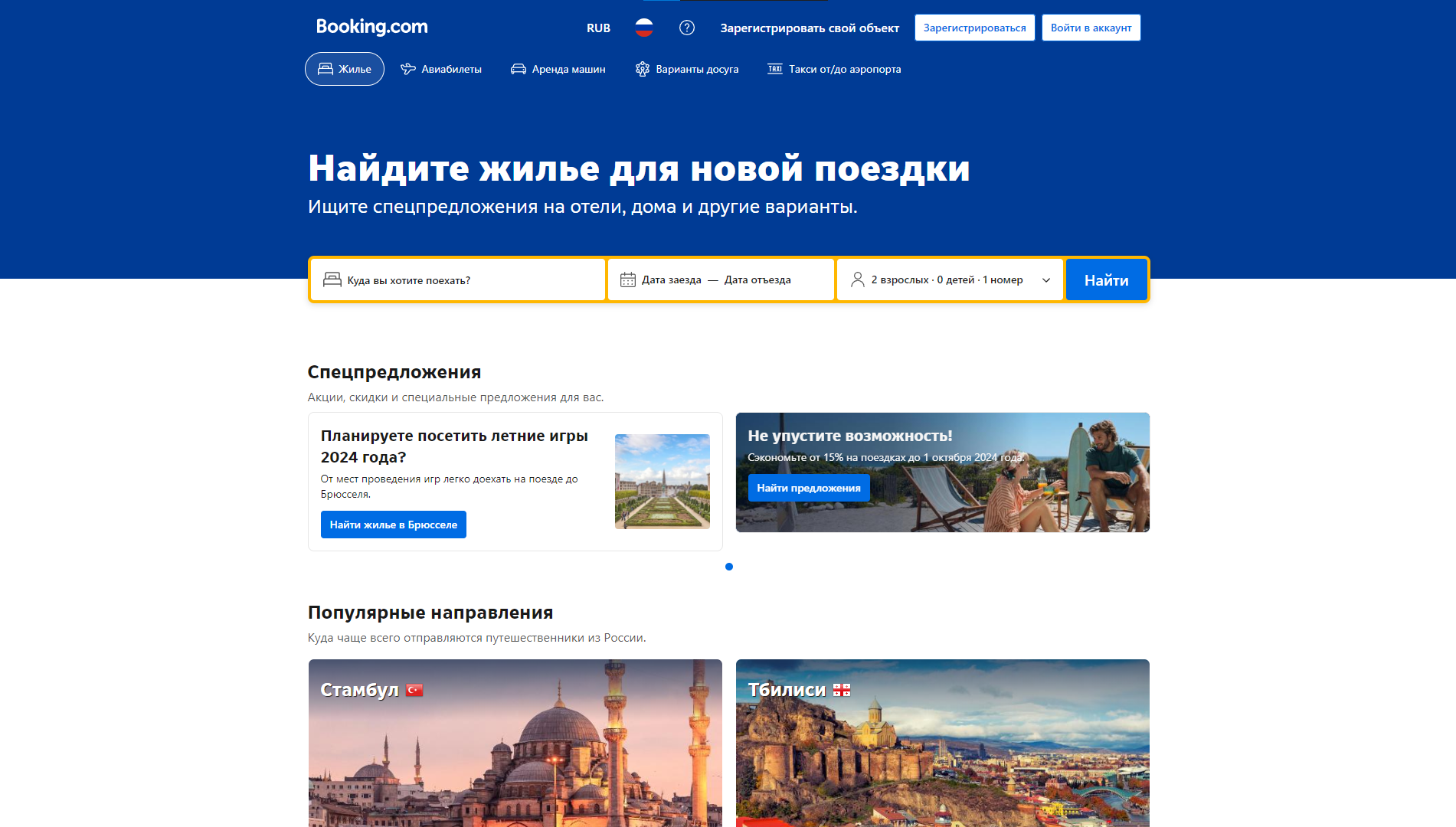
"Booking" является одним из наиболее известных и широко используемых онлайн-сервисов для бронирования гостиничного размещения. Этот сервис предлагает огромный выбор отелей, хостелов, апартаментов и других вариантов проживания по всему миру.

Достоинства:

* Обширная база отелей и различных типов жилья во множестве городов и стран;
* Подробные описания и отзывы о каждом объекте размещения от реальных гостей, что помогает пользователям принимать информированные решения;
* Гибкая система фильтрации и поиска, позволяющая настраивать запросы в соответствии с предпочтениями пользователей;

Недостатки:

* Некоторые пользователи могут столкнуться с проблемой избыточного выбора, что затрудняет процесс принятия решения;
* Возможность встретить неактуальную или недостоверную информацию о жилье, так как отзывы могут быть подвержены манипуляциям;
* Недостаток персонализации, поскольку сервис иногда предлагает одни и те же варианты для всех пользователей, не учитывая их индивидуальных предпочтений;
* Ушёл с российского рынка.



1. Главная страница сервиса Booking
   * 1. Airbnb

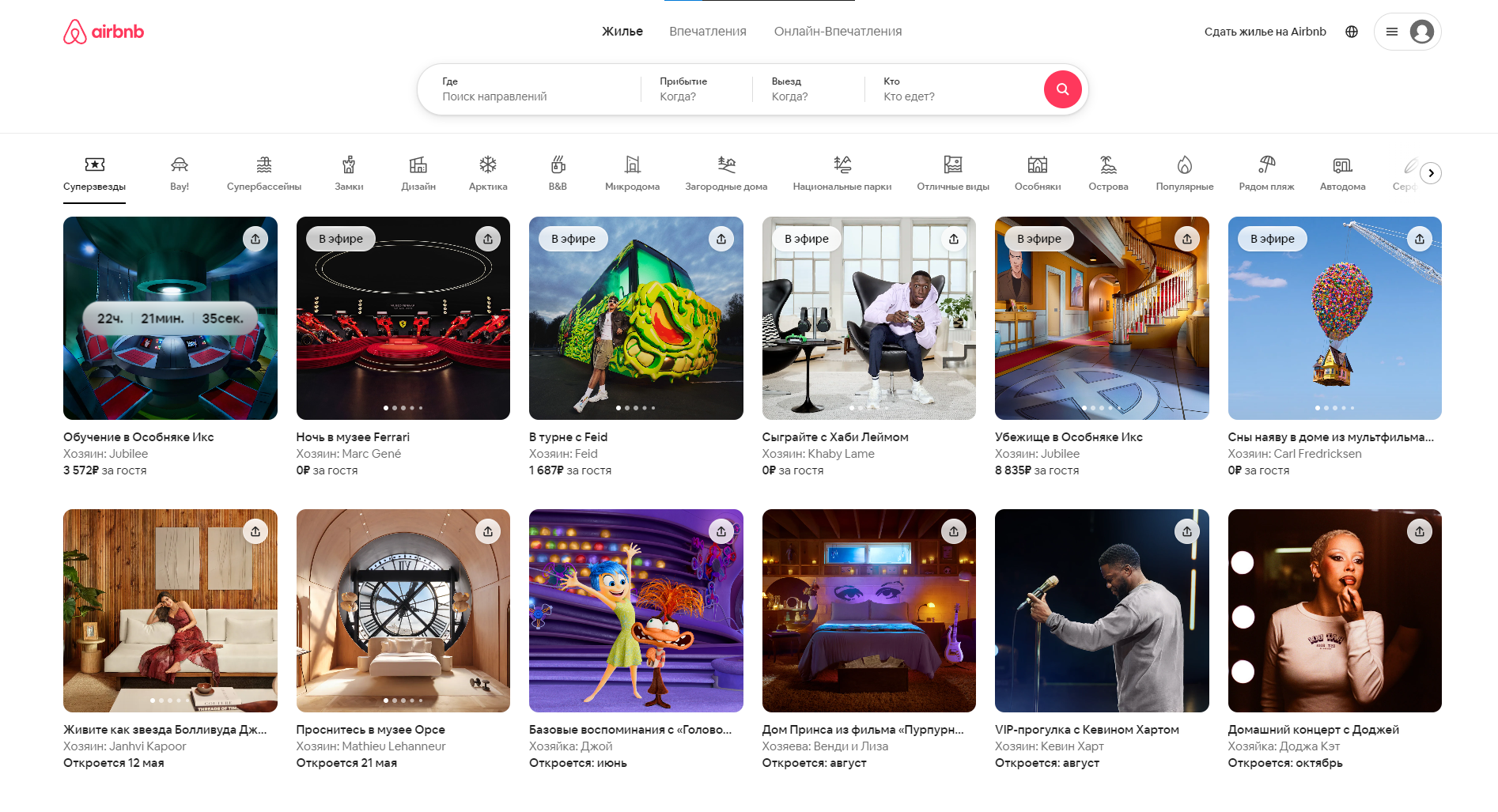
"Airbnb" представляет собой платформу для аренды жилья у местных жителей по всему миру. Этот сервис позволяет пользователям сдавать или арендовать жилье на короткий срок, включая квартиры, дома, комнаты и необычные объекты проживания.

Достоинства:

* Уникальные и разнообразные варианты размещения, включая дома в стиле "шале", деревенские уединенные уголки, маяки и другие нестандартные объекты;
* Возможность получить персональный опыт и советы от местных жителей, которые могут предложить рекомендации о местных достопримечательностях и культурных особенностях;
* Гибкая система ценообразования, позволяющая арендодателям и арендаторам договариваться о цене и условиях проживания;
* Интерактивные фильтры и карты для поиска жилья в определенном районе или близости к определенным объектам.

Недостатки:

* Необходимость общения и согласования с владельцем жилья, что может быть неудобно для тех, кто предпочитает более формальный и стандартизированный процесс бронирования;
* Риск непредвиденных обстоятельств или несоответствия фактического состояния жилья его описанию и фотографиям;
* Ушёл с российского рынка.



1. Главная страница сервиса Airbnb
   * 1. Яндекс.Путешествия

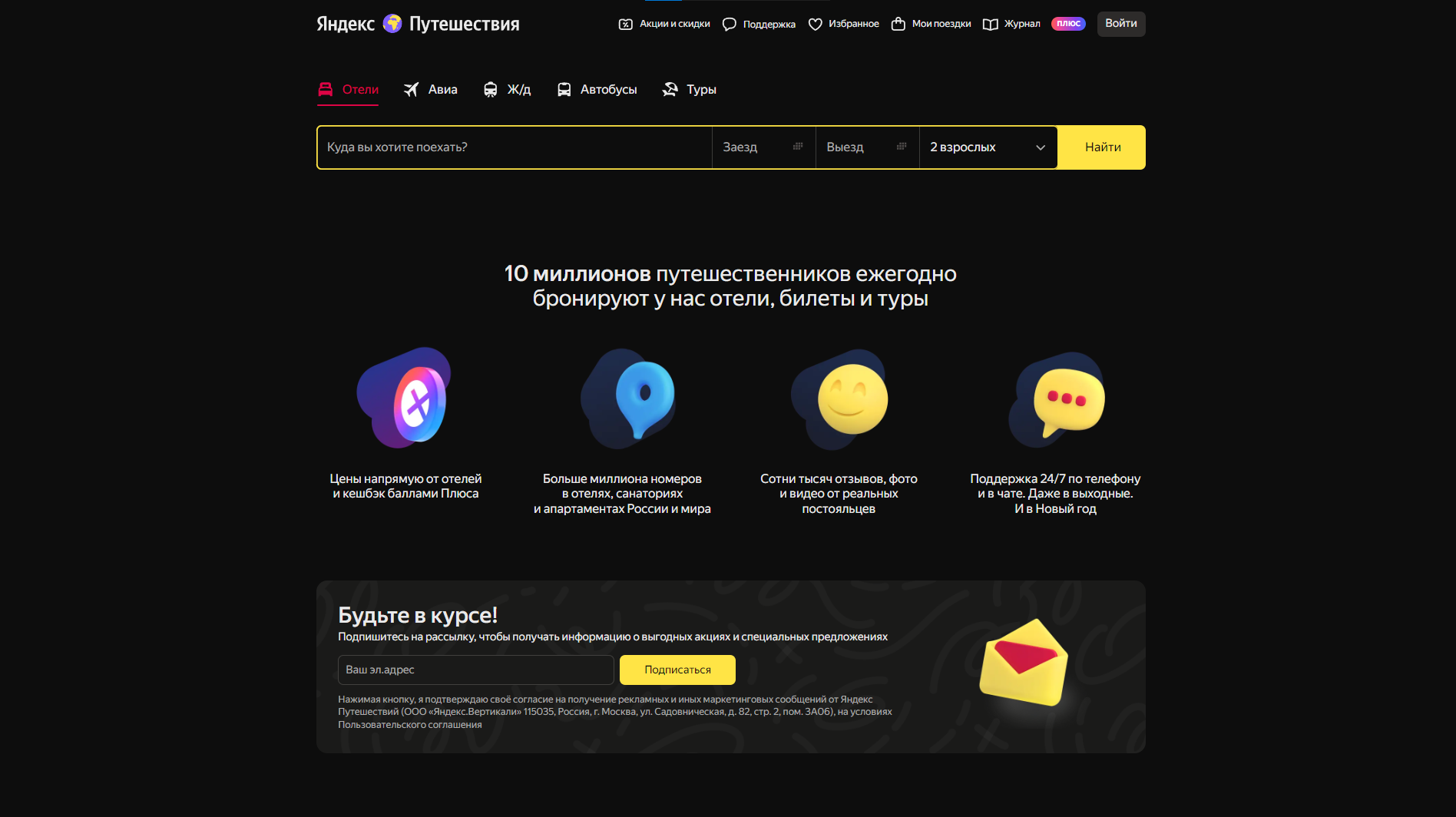
"Яндекс.Путешествия" представляет собой сервис для планирования и организации путешествий, который объединяет информацию о бронировании отелей, покупке авиабилетов, аренде автомобилей и других важных аспектах путешествия.

Достоинства:

* Интеграция с другими сервисами "Яндекса", такими как "Яндекс.Карты" и "Яндекс.Маршруты", что обеспечивает удобное планирование маршрутов и перемещений.
* Широкий выбор предложений от различных партнеров, включая отели, авиакомпании, агентства аренды автомобилей и турагентства;
* Удобный интерфейс и интуитивно понятная навигация, позволяющие быстро найти и забронировать необходимые услуги;
* Возможность получить скидки и специальные предложения для пользователей "Яндекс.Путешествий" от партнеров сервиса.

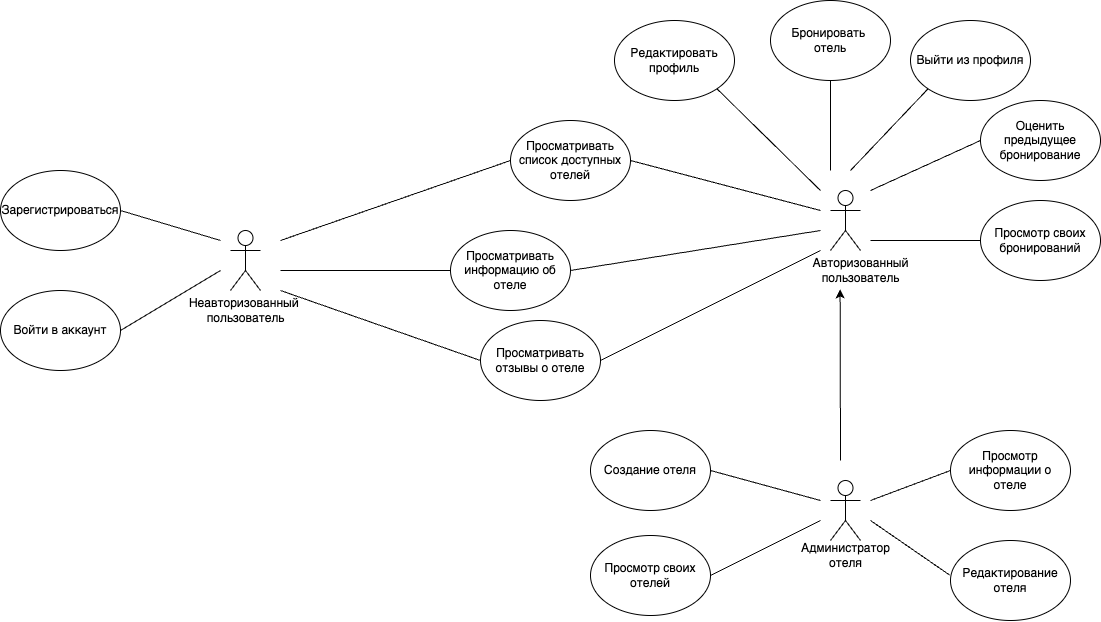
Недостатки:

* Ограниченная доступность информации и предложений по сравнению с некоторыми конкурентами, особенно в отношении международных путешествий и нестандартных видов размещения;
* Недостаточная персонализация и индивидуализация рекомендаций, что может привести к неоптимальному выбору для некоторых пользователей;
* Отсутствие собственного механизма обратной связи и отзывов пользователей о конкретных объектах размещения или услугах, что ограничивает возможности оценки и выбора.



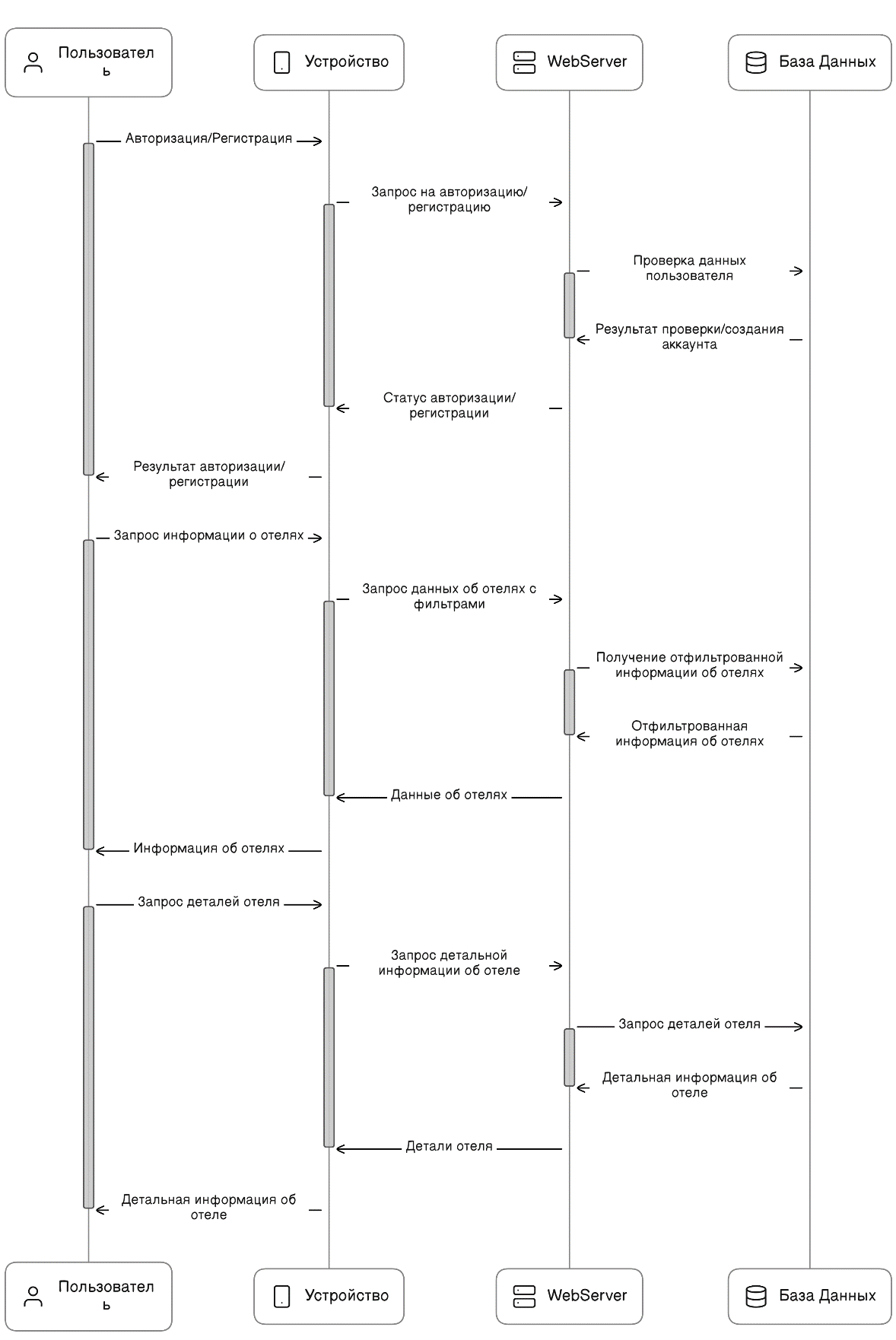
1. Главная страница сервиса Яндекс.Путешествия
2. Моделирование системы
   1. Диаграмма прецедентов

Рассмотрим полную диаграмму использования приложения различными типами пользователей (Рисунок 4). В данном контексте составление диаграммы прецедентов обусловлено необходимостью моделирования системы и понимания её функциональности и потребностей пользователей. Эти диаграммы помогают определить основные действия, которые пользователь должен совершить в системе для достижения определенных целей. Они также помогают выявить возможные риски и проблемы, которые могут возникнуть в процессе использования системы.

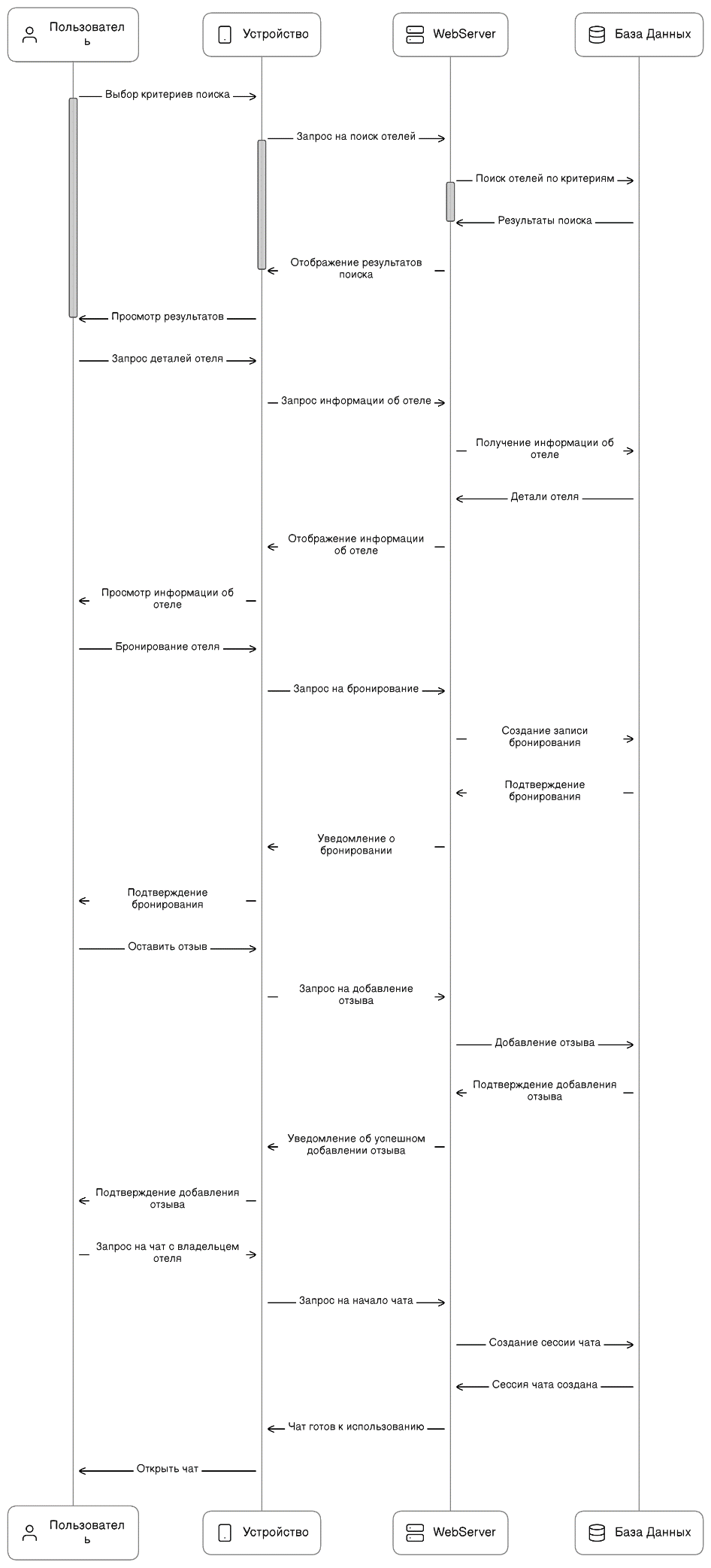


1. Диаграмма прецедентов (use-case)
   1. Диаграмма последовательности

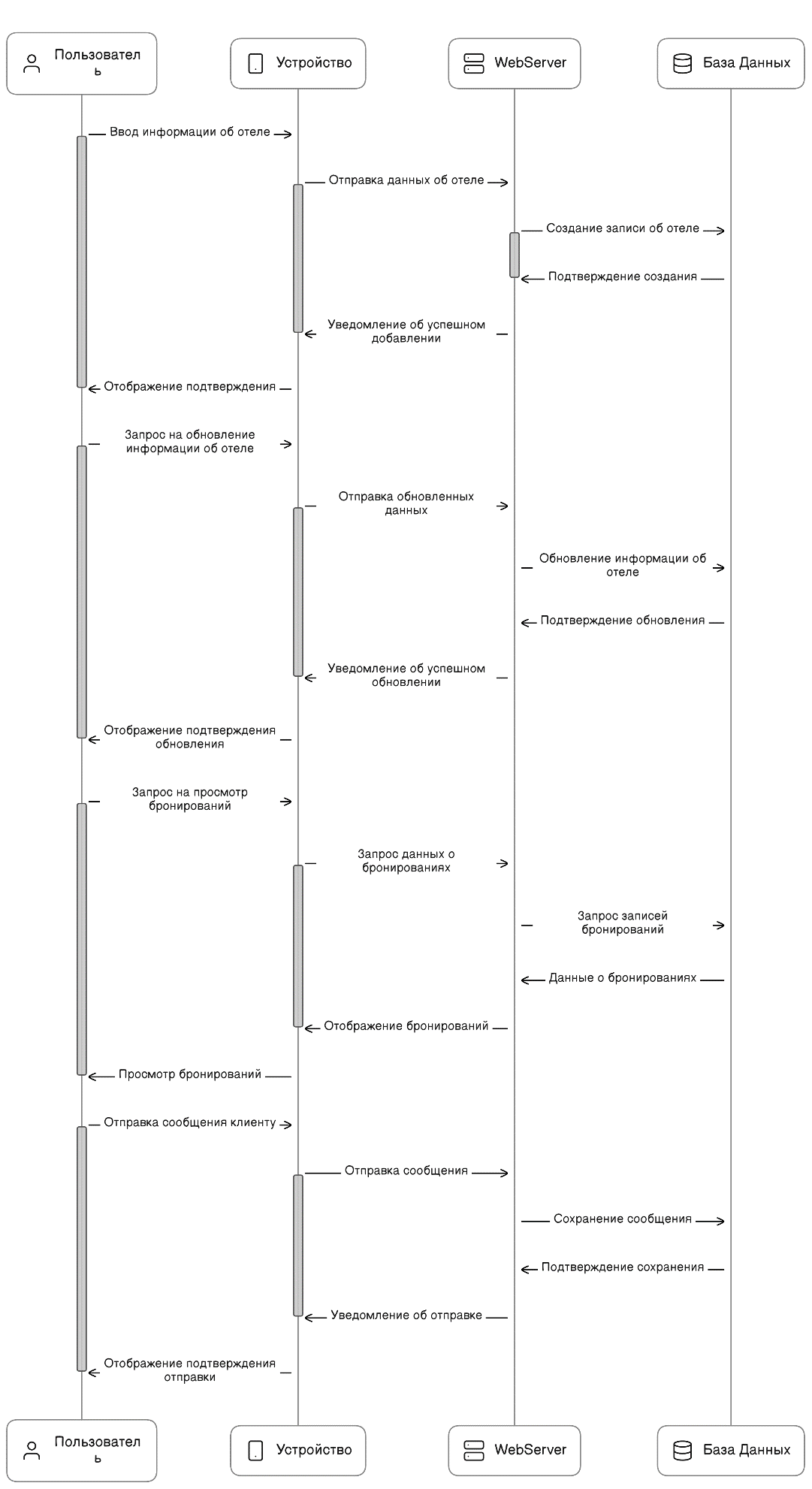
Диаграмма последовательности (Рисунки 5 - 7) играет важную роль в проекте, помогая более глубоко понять процесс, повысить его эффективность и упростить взаимодействие.



1. Диаграмма последовательности неавторизованного пользователя (sequence)

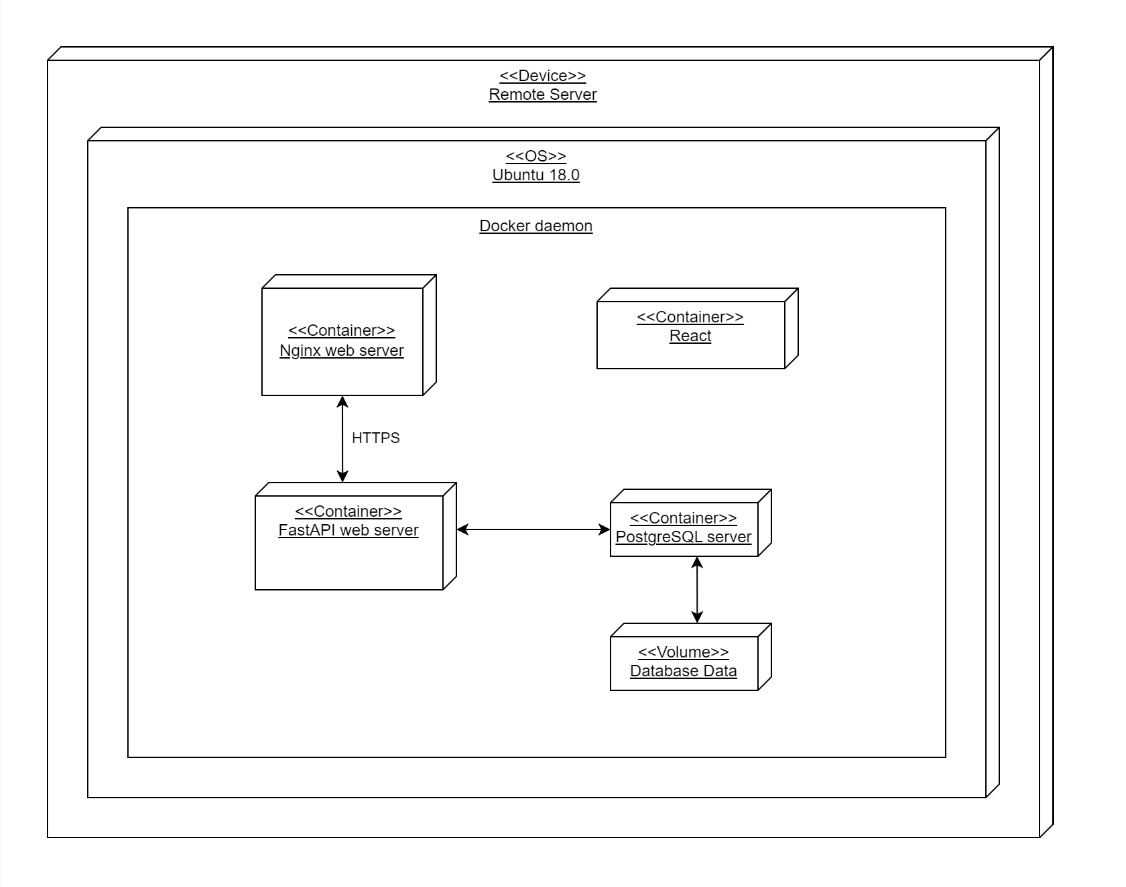


1. Диаграмма последовательности клиента (sequence)



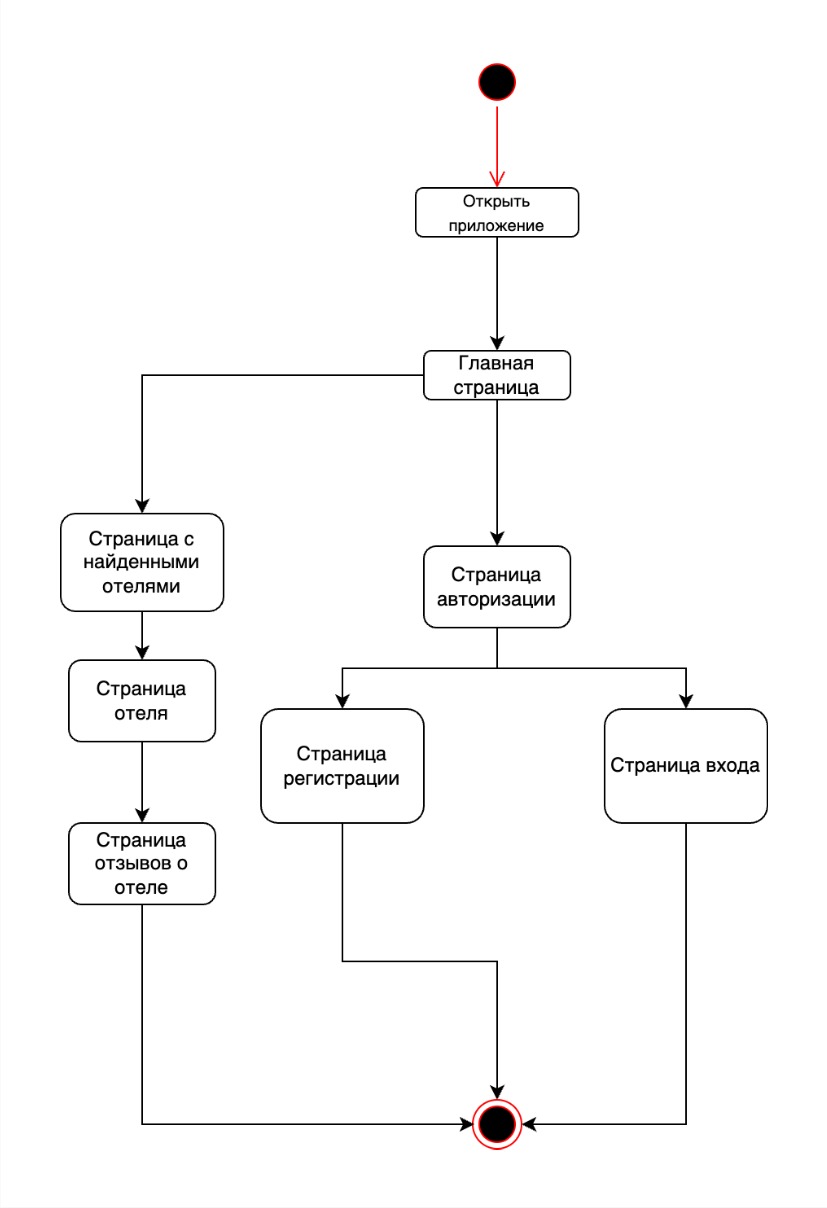
1. Диаграмма последовательности владельца отеля (sequence)
   1. Диаграмма развёртывания

Диаграмма развёртывания (Рисунок 8) помогает определить необходимости в аппаратном обеспечении, спланировать установку и настройку компонентов системы, а также оценить её производительность и масштабируемость.

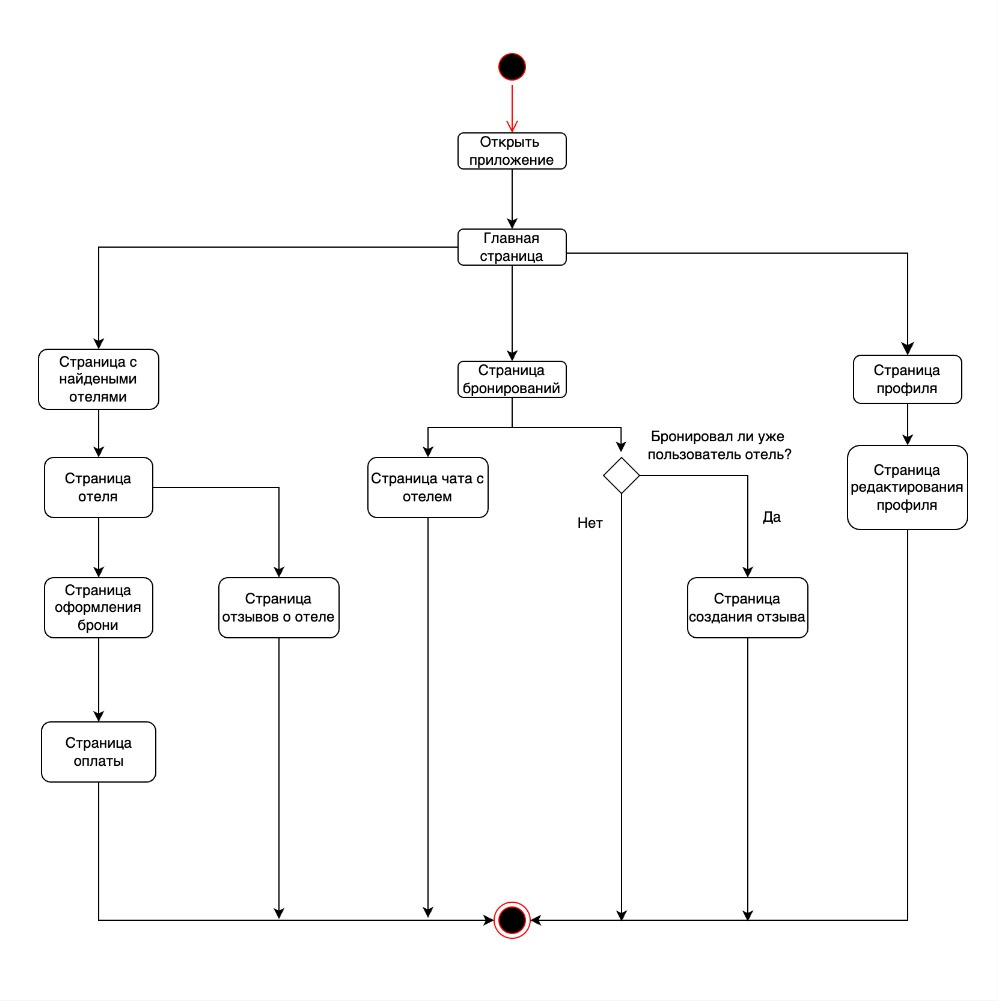


1. Диаграмма развертывания приложения
   1. Диаграммы состояния

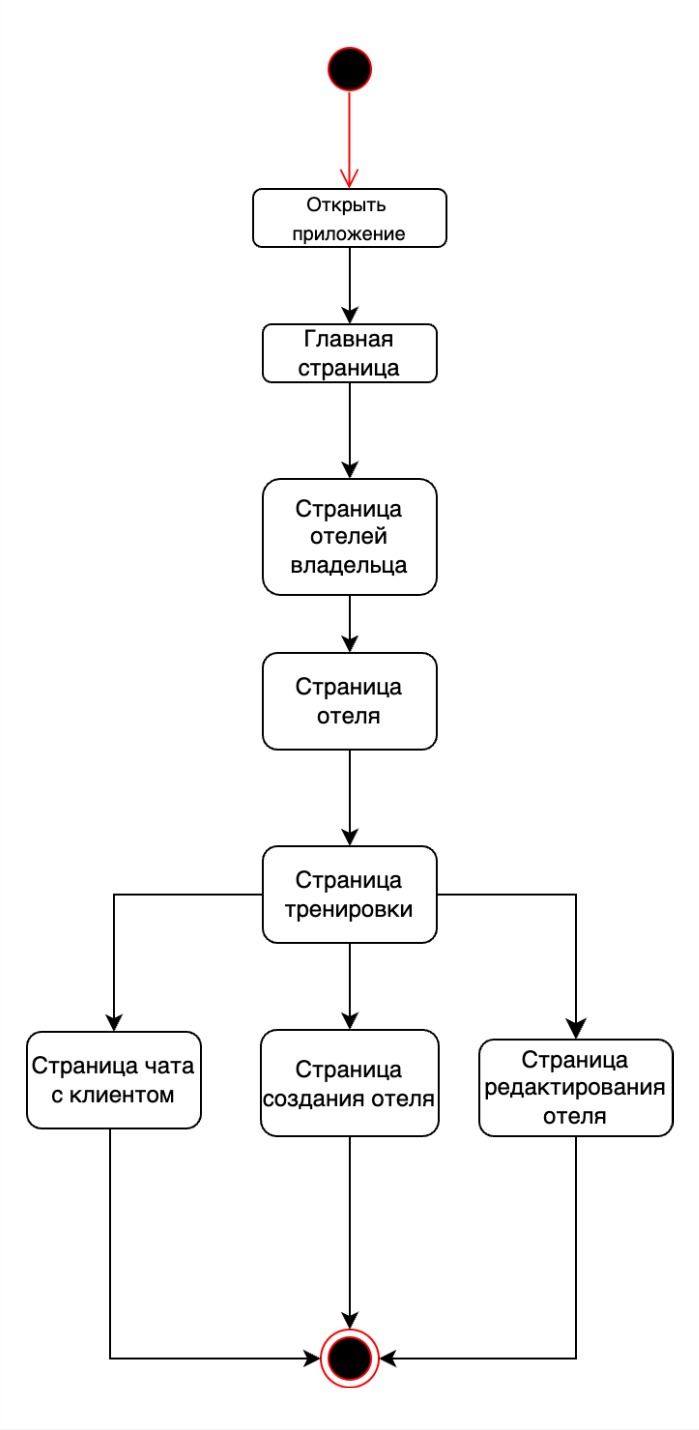
Диаграмма состояния (Рисунки 9 - 11) позволяет анализировать возможные сценарии поведения системы, выделять ключевые состояния и переходы между ними, а также оценивать её надежность и устойчивость к ошибкам. В рамках нашего проекта были разработаны три диаграммы состояний для неавторизованного пользователя, клиента и владельца отеля.



1. Диаграмма состояния неавторизованного пользователя (statechart)



1. Диаграмма состояния клиента (statechart)



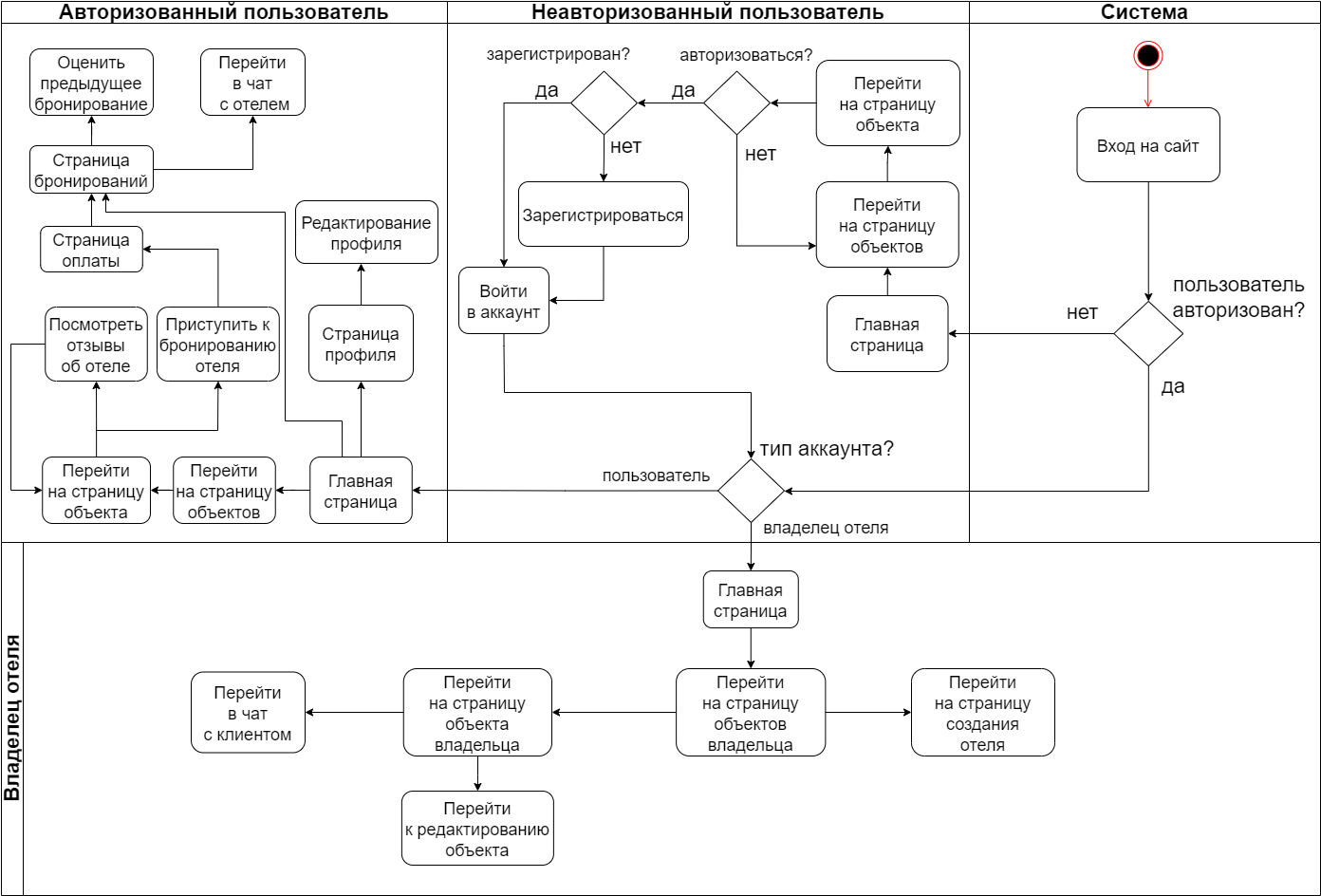
1. Диаграмма состояния владельца отеля (statechart)
   1. ER-диаграмма

ER-диаграмма (Рисунок 12) предоставляет структурное представление данных, иллюстрируя сущности (объекты) в системе и их взаимосвязи. Она помогает определить основные сущности, их атрибуты и отношения между ними, что облегчает проектирование базы данных и анализ требований к системе.



1. ER-диаграмма
   1. Диаграммы активности

Диаграмма активности (Рисунок 13) помогает улучшить понимание системных процессов, выявить и оптимизировать узкие места. Кроме того, она применяется для описания бизнес-процессов и управления проектами.



1. Диаграмма активности