ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет Высшая школа бизнеса
Образовательная программа “Бизнес-информатика”

Проект “Управление требованиями и проектирование ИС”

«Проектирование системы продажи и бронирования онлайн-билетов “Билеты-онлайн”»

Проект выполнили:
Шорин Матвей ББИ222, 33%
Статьев Александр ББИ222, 33%
Мухлаев Алан ББИ222, 33%

Москва,2024

Содержание проекта

Введение

Название проекта: “ Система покупки и бронирования онлайн билетов «Билеты-онлайн»”

Краткое описание разрабатываемой системы:

Система "Билеты онлайн" представляет собой платформу для автоматизации продажи и бронирования билетов на различные мероприятия, такие как концерты, кино, театральные постановки и выставки, организуемые заказчиком. Система предоставляет пользователям возможность поиска мероприятий, бронирования мест и оплаты билетов различными способами. Для каждого мероприятия доступен выбор категорий билетов, которые могут отличаться по цене и условиям.

Система предусматривает два типа пользователей: обычных и постоянных клиентов. Обычные пользователи могут приобретать билеты на отдельные мероприятия, в то время как постоянные клиенты, совершающие покупки регулярно, получают доступ к специальным скидкам и возможности бронировать большее количество билетов. Администратор системы управляет мероприятиями, обновляет информацию и следит за корректностью работы платформы, обеспечивая высокое качество обслуживания пользователей.

Для повышения удобства система интегрируется с различными платёжными системами и поддерживает уведомления пользователей о предстоящих мероприятиях.

Цель:

Разработать и внедрить к 1 сентября 2025 года систему онлайн-бронирования и продажи билетов, которая позволит пользователям легко находить мероприятия, бронировать и покупать билеты на концерты, театральные постановки, киносеансы и выставки. Система должна обеспечивать удобный интерфейс для поиска и выбора мероприятий, поддержку оплаты через различные платежные системы и предоставлять скидки для постоянных клиентов. Достичь уровня удовлетворенности пользователей не менее 85% в течение первых 6 месяцев после запуска, на основе опросов и отзывов пользователей.

Назначение:

Система предназначена для упрощения процесса бронирования и покупки билетов на мероприятия, автоматизации управления событиями и повышения эффективности обслуживания пользователей.

1. Список требований в форме технического задания.

1. Введение

1.1. Название проекта:

Система онлайн-бронирования и продажи билетов.

1.2. Цель проекта:

Разработать и внедрить к 1 сентября 2025 года систему онлайн-бронирования и продажи билетов, которая позволит пользователям легко находить мероприятия, бронировать и покупать билеты на концерты, театральные постановки, киносеансы и выставки. Система должна обеспечивать удобный интерфейс для поиска и выбора мероприятий, поддержку оплаты через различные платежные системы и предоставлять скидки для постоянных клиентов. Достичь уровня удовлетворенности пользователей не менее 85% в течение первых 6 месяцев после запуска, на основе опросов и отзывов пользователей.

1.3. Заказчик:

Название организации: ООО “КонцертТех”
Тип организации: Развлекательные услуги
Адрес фактический: 123 ул. Пушкина, Москва, Россия
Контактное лицо: Иван Иванович Петров
Электронная почта: petrov.ivan@concerttech.com
Телефон: +7 (777) 123-4567

1.4. Исполнитель:

Название фирмы: ООО “ТехСофт”
Тип фирмы: IT-компания
Адрес фактический: 456 ул. Программная, Москва, Россия
Контактное лицо: Шорин Матвей
Должность контактного лица: Генеральный директор
Электронная почта: shorin.matvey@techsoft.com
Телефон: +7(777) 213-7890

1.5. Дата начала проекта:

Вся работа над разработкой системы разделена на несколько стадий. После каждого этапа исполнитель обязан составить отчет о результатах работы:

- Анализ требований и проектирование 01.10.2024 – 01.12.2024
- Разработка 01.12.2024 – 01.04.2025
- Тестирование 01.04.2025 – 01.07.2025
- Внедрение 01.07.2025 – 01.09.2025
- Сопровождение и обновление: с 01.09.2025

Плановые сроки завершения разработки и внедрения проекта: 01.09.2025

2. Общее описание

2.1. Общий взгляд на продукт:

Продукт представляет собой **систему онлайн-бронирования и продажи билетов**, предназначенную для организации и управления продажами билетов на различные мероприятия, такие как концерты, киносеансы, театральные постановки и выставки. Продукт предлагает интуитивно удобный пользовательский интерфейс, который позволяет пользователям находить интересующие их мероприятия. Продукт будет интегрироваться с платежными системами и обеспечивать высокий уровень безопасности для защиты персональных данных пользователей и проведения транзакций. Основной упор сделан на удобство использования, производительность и безопасность.

Основная функция системы заключается в предоставлении пользователям возможности:

- Просматривать доступные мероприятия;
- Использовать фильтры для поиска по дате, месту, времени и цене;
- Покупать билеты с поддержкой разных методов оплаты;
- Получать электронные билеты сразу после покупки на электронную почту и в личный кабинет;
- Использовать систему лояльности для постоянных клиентов.

2.2. Классы и характеристики пользователей:

- **Обычные пользователи**:
    - Это пользователи, которые покупают билеты время от времени. Их основная задача — быстро и просто найти нужное мероприятие, выбрать подходящее место и оплатить билеты. Для них важны скорость работы системы, удобный интерфейс и поддержка различных методов оплаты.
    - **Характеристики**: низкий уровень постоянного взаимодействия с системой, основной интерес — разовые и нечастые покупки билетов. Возраст: 18-45 лет.

- **Постоянные клиенты**:
    - Это пользователи, которые часто покупают билеты через систему (зачастую семьи). Они заинтересованы в получении скидок через систему лояльности. Для них важна возможность быстрой покупки нескольких билетов на разные мероприятия, а также возможность бронирования билетов на несколько мероприятий сразу.
    - **Характеристики**: высокая частота использования, постоянное взаимодействие с системой, заинтересованность в дополнительных опциях (например, участие в системе лояльности). Возраст: 25-80 лет.
- **Администраторы системы**:
    - Это сотрудники, которые управляют событиями, обновляют информацию о мероприятиях, следят за корректностью работы системы, а также решают проблемы пользователей.
    - **Характеристики**: высокий уровень взаимодействия с системой, доступ к административным функциям для управления мероприятиями и изменения доступных клиентам статусов в системе лояльности.

2.3. Среда функционирования

Приложение "Билеты онлайн" будет работать в мобильной и веб-версиях, обеспечивая доступность и удобство работы для всех категорий пользователей.

Основные компоненты среды функционирования:

- **Операционные системы**:
    - Система должна поддерживать работу в различных операционных системах, как на стороне клиента, так и на сервере:
        - **Клиентская часть**:
            - **Мобильное-приложение**: Основная версия продукта будет доступна через мобильное приложение. Это позволит пользователям совершать покупку билетов с мобильных устройств и планшетов. Приложение будет разработано для операционных систем **iOS** (версии 16 и выше) и **Android** (версии 8.0 и выше), оно будет иметь адаптивный дизайн для удобного использования на различных экранах.
            - **Веб-версия**: Сервис также будет доступен через веб-браузеры. Это позволит пользователям совершать покупку билетов с настольных компьютеров, ноутбуков и мобильных устройств, используя такие браузеры, как **Google Chrome**, **Safari и др.**
        - **Серверная часть**: Рекомендуется использовать серверы на базе **Linux** или **Windows Server**, поддерживающие стабильность и безопасность.

    - **Сетевое окружение**: Система должна быть доступна через интернет и поддерживать работу в условиях высоких и низких скоростей соединения. Важно, чтобы пользователи с медленным интернет-соединением могли корректно использовать систему.

Особенности среды функционирования:

- **Мобильность**: Важна полноценная поддержка мобильных, переносных и стационарных устройств, чтобы была возможность приобрести или забронировать билеты как на телефоне/планшете, так и на компьютере или ноутбуке.
- **Доступность и масштабируемость**: Система должна быть доступна круглосуточно и поддерживать возможность быстрой масштабируемости при увеличении числа пользователей или мероприятий.
- **Безопасность**: Операционная среда должна обеспечивать высокий уровень защиты персональных данных, включая шифрование, защиту от атак и управление правами доступа.

2.4. Ограничения реализации

Ограничения реализации системы включают факторы, которые могут повлиять на ее разработку, работу и дальнейшее расширение.

2.4.1. Ограничения по производительности

- Система должна обрабатывать большое количество запросов в пиковые моменты (например, перед массовыми мероприятиями). Это требует оптимизации кода.
- Для поддержки массовых продаж билетов, сервер должен иметь достаточные мощности для обработки до нескольких сотен транзакций в минуту.

2.4.2. Ограничения по безопасности

- Обработка персональных данных пользователей (например, данные при регистрации) и финансовых данных требует соблюдения стандартов защиты, таких как **PCI DSS** для безопасных транзакций и **GDPR** для защиты персональных данных.
- Необходимо обеспечить регулярное обновление и проверку безопасности системы, чтобы избежать утечек данных и атак.

2.4.3. Ограничения по совместимости

- Интерфейс и функционал системы должны быть полностью совместимы с различными веб-браузерами и операционными системами (как настольными, так и мобильными).

- Возможны ограничения по работе на устаревших версиях браузеров и мобильных операционных системах.

2.4.4. Ограничения по срокам разработки

- Проект имеет ограниченные сроки для разработки минимально жизнеспособного продукта, что может привести к необходимости упрощения или отсрочки внедрения некоторых функций на начальном этапе.
- Возможны ограничения по ресурсам команды разработки, что повлияет на выбор технологий и архитектурных решений для ускорения разработки.

3. Ключевые функциональные требования

3.1. Регистрация и личный кабинет

- **Регистрация пользователей**:
    - Пользователи должны иметь возможность зарегистрироваться разными способами:
        - Email: пользователь вводит свой email-адрес и создает пароль.
        - Номер телефона: пользователь вводит свой номер телефона и с помощью SMS получает код подтверждения, введя который завершает регистрацию.
        - Социальные сети (ВК и другие): Пользователь может зарегистрироваться в приложение с помощью существующих аккаунтов в социальных сетях.
        - В процессе регистрации пользователи заполняют обязательные и опциональные поля с данными.
          Обязательные:
          1. Номер телефона или email.
          2. Пароль.
          Опциональные:
          1. ФИО (при использовании регистрации через соц. сети данное поле заполняется автоматически).
          2. Дата рождения.
          3. Город.
- **Личный кабинет пользователя**:
    - В личном кабинете будут храниться личные данные, такие как имя, фамилия, контактная информация.
    - Личный кабинет также будет содержать историю покупок, статус пользователя (обычный или постоянный клиент), доступ к скидкам для постоянных клиентов и информацию о действующих билетах.

    - В личном кабинете будет возможно изменить регистрационные данные аккаунта, такие как пароль, email или телефон.

3.2. Поиск мероприятий и фильтрация

- **Фильтры для поиска**:
    - Пользователи смогут искать мероприятия по нескольким критериям: тип мероприятия (концерты, кино, театр), дата, место проведения, стоимость билетов.
    - Система должна поддерживать сортировку по популярности мероприятий, новизне, цене и категориям.
- **Отображение результатов поиска**:
    - Результаты поиска будут представлены в виде карточек мероприятий с краткой информацией (название, дата, место, стоимость билетов).
    - При нажатии на карточку мероприятия пользователь перейдет на страницу с детальной информацией о событии.
- **Алгоритм подбора:**
    - Подбор на основе предпочтений: Приложение анализирует предпочтения пользователя и предлагает ему более подходящие мероприятия.
    - Приоритеты поиска: Пользователям при поиске новых мероприятий, будут предлагаться новинки и наиболее популярные мероприятия сейчас.
    - Возможность пометить рекомендуемые события как неинтересные.
- **Дополнительные функции поиска**:
    - История поиска: Приложение будет сохранять последние результаты, чтобы пользователь мог быстро повторить поиск.
    - Возможность пометить мероприятие как “Понравившееся”, и оно отобразиться во вкладке “Понравившееся”.

3.3. Категории мероприятий:

Система разделяет мероприятия по категориям, что позволяет пользователям легко находить активности по интересам.

Примеры категорий:

- ТОП-10: Алгоритм отслеживает мероприятия по количеству купленных билетов и заносит их в данную категорию
- Кино: Киносеансы
- Концерты: Выступления музыкальных исполнителей
- Театр: Различного рода постановки (например, спектакли, мюзиклы и другие)
- Детям: Мероприятия с пометкой 0+ и 6+
- Выставки: ярмарки, музеи и выставки, посвященные разным темам

    - Спорт: Матчи разных видов спорта (например, футбол, баскетбол и другие)
    - Стендап: Stand-up выступления звезд, комиков.

3.4. Страница мероприятия

- **Информация о мероприятии**:
    - Полное описание мероприятия, включая дату, место проведения, длительность, описание программы, цены на билеты.
    - Визуальная карта зала с возможностью выбора конкретного места (если предусмотрено мероприятием).
- **Система отзывов**:
    - На странице мероприятия необходимо предоставить возможность написать отзыв о мероприятии, а также прочитать уже написанные другими пользователями отзывы.
- **Кнопка "Купить билет"**:
    - Пользователь может выбрать количество билетов(в зависимости от статуса) и категории (обычные или VIP).
    - После выбора система предложит доступные методы оплаты.

3.5. Оформление покупки

- **Выбор способа оплаты**:
    - Пользователи могут выбрать один из доступных способов оплаты: банковская карта, электронные кошельки (например, ЮMoney, СБП, Е-касса).
- **Подтверждение покупки**:
    - После успешной оплаты пользователю на электронную почту отправляется подтверждение покупки и электронный билет.
    - В личном кабинете пользователя также доступен электронный билет, который можно скачать или распечатать.

3.6. Система лояльности для постоянных клиентов

- **Профили постоянных клиентов**:
    - Постоянные клиенты (клиенты, совершившие более 10 покупок) получают дополнительные привилегии, такие как доступ к скидкам и покупка до 6 билетов за один раз на одно мероприятие.

3.7. Поддержка пользователей

- **Система помощи и обратной связи**:
    - В системе будет реализован раздел часто задаваемых вопросов (FAQ) и контактная форма для обратной связи с поддержкой.

    - Пользователи смогут оставить заявку на возврат билета или сообщить о проблеме с покупкой в чате с поддержкой.

4. Нефункциональные требования

Нефункциональные требования описывают параметры системы, которые обеспечивают ее стабильную и эффективную работу, а также удобство использования.

4.1. Производительность и отклик системы

- Скорость отклика системы: Время отклика системы должно составлять не более 2 секунд при взаимодействии с основными функциями (поиск мероприятий, просмотр страницы события, покупка билета). Это включает и работу в условиях повышенной нагрузки.
- Обработка большого числа запросов: Система должна поддерживать одновременную работу не менее 10 000 пользователей, особенно в пиковые моменты, такие как массовая продажа билетов на популярные мероприятия.
- Масштабируемость: При увеличении числа пользователей или мероприятий система должна масштабироваться горизонтально (путем добавления серверных мощностей) без ухудшения производительности.

Механизмы оптимизации:

- Кэширование: Применение системы кэширования с Redis для временного хранения часто используемых данных, таких как профили пользователей, списки интересов и настройки.
- Балансировка нагрузки: Внедрение балансировщиков нагрузки, например, Nginx или HAProxy, для равномерного распределения запросов между серверами, что поможет избежать перегрузки отдельных компонентов.

4.2. Масштабируемость и гибкость

- Масштабирование серверных ресурсов: Система должна быть спроектирована таким образом, чтобы можно было легко увеличить количество серверов и базы данных в случае необходимости обработки увеличенного количества пользователей или транзакций.
- Поддержка новых функций: Архитектура системы должна предусматривать возможность легкого добавления новых функциональных модулей без необходимости внесения серьезных изменений в основную структуру.

4.3. UX/UI

- **Интуитивно понятный интерфейс**: Интерфейс должен быть максимально простым и интуитивно понятным для пользователей, с минимальным количеством шагов для выполнения основных операций (поиск мероприятия, выбор места, покупка билета).
- **Адаптивный дизайн**: Приложение должно поддерживать адаптивный дизайн, чтобы корректно отображаться на разных устройствах (мобильные телефоны, планшеты, настольные компьютеры) и в разных браузерах.
- **Многоязычная поддержка**: Приложение должно поддерживать несколько языков интерфейса (например, русский и английский), что позволит расширить аудиторию.
- **Персонализация:**Возможность пользователей настраивать цветовые схемы приложения и фильтровать только интересующие их события.
- **Уведомления:** Система уведомлений, реализованная для мобильных приложений на операционных системах IOS и Android, позволит пользователям узнавать о новых мероприятиях или интересных подборках через push-уведомления.

4.4. Кроссплатформенность

Приложение "Билеты онлайн" будет кроссплатформенным, что позволит пользователям взаимодействовать с системой с различных устройств и операционных систем.

- **Мобильные устройства**: Приложение должно поддерживать работу на операционных системах **iOS** и **Android** . Мобильные версии должны обладать полным функционалом веб-приложения.
- **Веб-приложение**: Доступ через веб-браузеры должен быть обеспечен для всех популярных браузеров, включая **Google Chrome**, **Mozilla Firefox**, **Safari** и **Microsoft Edge**. Приложение должно поддерживать последние версии этих браузеров.
- **Синхронизация:**Ключевым аспектом является обеспечение синхронизации данных между мобильной и веб-версиями, чтобы пользователи могли беспрепятственно переключаться между устройствами, сохраняя всю информацию.

4.4.1 Преимущества кроссплатформенного подхода:

- **Единый пользовательский опыт**: Независимо от того, какое устройство используют пользователи (мобильное или настольное), интерфейс и функциональность будут схожими, что облегчит освоение системы.
- **Поддержка большого числа устройств**: Кроссплатформенная разработка обеспечит доступ к системе для максимального числа пользователей, что важно для привлечения широкой аудитории.

4.5. Надежность

- **Безотказная работа**: Система должна обеспечивать минимальный процент сбоев и простоев, менее 1% в год, что соответствует уровню доступности **99%** времени.
- **Устойчивость к ошибкам**: В случае возникновения ошибок система должна обеспечить минимальное влияние на пользователей и продолжать работу с минимальными задержками.

5.Обязательные и опциональные требования

Этот раздел включает требования, которые являются критически важными для успешной работы системы, а также опциональные функции, которые могут быть внедрены для улучшения пользовательского опыта и расширения возможностей системы.

5.1. Обязательные требования

1. **Регистрация и аутентификация пользователей**:
    - Поддержка регистрации через email, телефон, а также авторизация через социальные сети (Например:Вконтакте).
    - Возможность восстановления пароля через электронную почту или SMS.
        i. Пользователь вводит свой зарегистрированный email или номер телефона на странице восстановления пароля.
        ii. Система отправляет одноразовый код подтверждения на указанный контактный метод (email или SMS).
        iii. После ввода кода подтверждения пользователю предоставляется возможность создать новый пароль.
2. **Личный кабинет пользователя**:
    - Возможность редактирования личных данных (имя, контактная информация, город).
    - Просмотр истории покупок,скачивание электронных билетов и информацию о действующих билетах.
    - Отображение бонусов и скидок для постоянных клиентов.
3. **Поиск и фильтрация мероприятий**:
    - Возможность поиска мероприятий по категориям (концерты, кино, театр, выставки) и фильтрации по дате, месту проведения, цене и типу билетов.
    - Сортировка мероприятий по популярности, новизне, цене и дате проведения.
4. **Покупка билетов**:
    - Выбор места в зале (если предусмотрено мероприятием) через визуальную карту зала.

    - Поддержка различных способов оплаты: банковские карты, электронные кошельки (ЮMoney, СБП).
    - Отправка электронных билетов на email пользователя после завершения покупки.
5. **Безопасность данных**:
    - Шифрование всех передаваемых данных с использованием SSL/TLS.
    - Соответствие стандартам **PCI DSS** для обработки платежей.
    - Регулярные обновления безопасности и аудиты.
6. **Система лояльности**:
    - Доступ к специальным предложениям и скидкам для постоянных клиентов.
    - Покупка до 6 билетов за один раз на одно мероприятие.
7. **Интеграция с внешними системами**:
    - Подключение к внешним API для автоматического обновления информации о мероприятиях.
    - Интеграция с платежными шлюзами для поддержки разных способов оплаты.
8. **Уведомления**:
    - Отправка push-уведомлений о новых мероприятиях, интересных подборках и тд. пользователям.

5.2. Опциональные требования

1. **Персонализированные рекомендации**:
    - Использование алгоритмов для анализа предпочтений пользователей и предложений мероприятий, которые могут им понравиться на основе предыдущих покупок и действий.
2. **Поддержка видеоконтента**:
    - Возможность добавления видеоматериалов к странице мероприятия (трейлеры фильмов, презентации концертов).
3. **Продажа дополнительных услуг**:
    - Возможность покупки дополнительных услуг (например, бизнес-ланчи, участие в мастер-классах) вместе с билетами на мероприятие.
4. **Интеграция с календарями**:
    - Автоматическая интеграция с календарями пользователей (Google Calendar, Apple Calendar), чтобы добавить купленные билеты и напоминания о мероприятиях.

5.3. Обоснование опциональных требований

Опциональные требования добавляют ценность системе, но не являются критически важными для первоначального запуска (MVP). Их реализация может быть отложена до дальнейших этапов разработки, когда будет достигнут стабильный рост пользователей. Например:

- **Поддержка видеоконтента**: Возможность просмотра видеороликов или трейлеров мероприятий на странице события может быть введена для увеличения вовлеченности пользователей.

6. Безопасность данных

- **Защита персональных данных**:
    - Все данные пользователей должны быть защищены с помощью шифрования SSL/TLS.
    - Вход в систему и осуществление покупок требуют обязательной аутентификации.
    - Хранение паролей в зашифрованном виде (bcrypt).
      Это гарантирует, что даже в случае утечки базы данных пароли останутся защищенными.
- **Защита транзакций**:
    - Платежи через систему должны соответствовать стандартам PCI DSS для обеспечения безопасности финансовых операций.
- **Шифрование данных**:
    - Платежные данные не должны храниться на серверах приложения без специального разрешения от пользователя. В случае сохранения данных — необходимо использование безопасных протоколов и стандартов PCI DSS.
- **Аутентификация и авторизация:**
    - Ограничение количества попыток входа: После нескольких неудачных попыток входа в систему учетная запись может временно блокироваться для предотвращения атак методом подбора паролей.
    - Восстановление пароля: При утери пароля, пользователь сможет восстановить его через телефон или email.
- **Защита от атак:**
    - DDoS-защита: Применение защитных решений, таких как WAF (Web Application Firewall), для предотвращения атак отказа в обслуживании.
- **Политика конфиденциальности и согласие на обработку данных:**
    - В системе будет реализована подробная политика конфиденциальности, в которой будет указано, какие данные собираются, как они обрабатываются, хранятся и используются. Пользователи смогут ознакомиться с этой политикой перед регистрацией.
    - При регистрации пользователи будут обязаны подтвердить согласие на обработку своих персональных данных в соответствии с требованиями **общего регламента по защите данных.**
- **Обновления и проверки безопасности:**

- **Плановые обновления безопасности**: Система будет регулярно обновляться для устранения выявленных уязвимостей и улучшения безопасности данных. Это включает обновления серверного и клиентского программного обеспечения.

7. Приоритизация требований по методу MoSCoW.

В качестве обязательных и опциональных требований мы выделили и обобщили самые основные для более простого понимания ключевых задач нашего ПО. Для приоритизации мы использовали метод MoSCoW, так как он помогает определить, какие требования к проекту являются наиболее важными и должны быть реализованы в первую очередь, а какие могут быть отложены или рассмотрены в последующих итерациях. Метод MoSCoW помогает лучше понимать приоритеты и наиболее важные элементы проекта как заказчику, так и команде проекта.

Обязательные и опциональные требования:

Must Have(Должно быть):

- **Регистрация и аутентификация:** Обязательно предоставить систему регистрации и аутентификации для пользователей, чтобы обеспечить безопасность данных и контроль доступа.
- **Поиск и фильтрация мероприятий:** Обязательно должна быть возможность поиска мероприятий по дате, месту проведения, типу и стоимости билетов, а также фильтрация мероприятий по запросу пользователя.
- **Отправка электронных билетов:** Электронные билеты должны отправляться на email пользователя после успешной оплаты.
- **Покупка билетов и оплата**: Поддержка различных способов оплаты (банковские карты, электронные кошельки, СБП).
- **Личный кабинет пользователя**: Пользователь должен иметь доступ к личному кабинету с историей покупок, статусом пользователя, информацией о действующих билетах.
- **Служба поддержки**: Обязательно должна быть связь со службой поддержки для решения проблем с возвратами билетов и неисправностями системы.

Should Have(Желательно):

- **Уведомления**: Внедрение системы уведомлений о предстоящих мероприятиях, скидках и акциях через push-уведомления .

- **Отзывы**: Пользователи должны иметь возможность оставлять отзывы и оценки для мероприятий.
- **Персонализированные рекомендации**: Система должна предлагать мероприятия на основе интересов пользователя, предыдущих покупок и действий.

Could Have(Могло бы быть):

- **Поддержка видеоконтента**: Возможность просмотра видеороликов или трейлеров мероприятий на странице события.
- **Интеграция с социальными сетями**: Возможность делиться информацией о мероприятиях через социальные сети и приглашать друзей на события.
- **Аналитика и отчетность**: Внедрение модуля аналитики для анализа поведения пользователей и эффективности мероприятий.

Won’t Have(Не должно быть):cc

- **Интеграция с внешними системами бронирования**: Интеграция с другими платформами для бронирования мероприятий не является приоритетной на данном этапе.

2. Модель жизненного цикла и методология разработки ПО

2.1. Модель жизненного цикла программного обеспечения

Для проекта «Билеты онлайн» была выбрана гибкая модель жизненного цикла, сочетающая инкрементный подход с методологией Agile. Это решение обеспечивает постепенное расширение функциональности и оперативную адаптацию к изменениям требований. Такой выбор был сделан с учётом специфики проекта, где важны частые обновления продукта, отзывчивость на запросы пользователей и плавная интеграция с внешними сервисами, включая платежные системы.

2.1.1. Основные преимущества инкрементной модели для проекта «Билеты онлайн»

1. **Постепенное внедрение функционала**: Инкрементная модель позволяет поэтапно добавлять новые возможности в систему, обеспечивая раннее тестирование и использование готовых частей продукта. Например, на начальном этапе можно реализовать базовую функциональность бронирования билетов, а на последующих — интеграцию с платежными системами и модулями для работы с клиентами.
2. **Эффективное управление рисками**: Каждый инкремент системы подлежит тестированию и проверке, что позволяет своевременно выявлять и минимизировать потенциальные риски. Это особенно важно для проекта «Билеты онлайн», который работает с персональными данными и финансовыми операциями.
3. **Гибкость**: Инкрементная модель позволяет легко вносить изменения на любом этапе разработки. По мере появления новых требований, связанных, например, с пользовательским интерфейсом или добавлением статусов лояльных клиентов, их можно интегрировать в следующих инкрементах.

2.1.2. Основные преимущества agile-подхода

1. **Постоянное взаимодействие с заказчиком**: Agile позволяет регулярно общаться с заказчиком и демонстрировать результаты работы, что снижает риск несоответствия ожиданиям. В случае проекта «Билеты онлайн» это означает возможность получать обратную связь от пользователей и адаптировать функционал под реальные потребности.
2. **Гибкое планирование и быстрые итерации**: Спринты по методологии agile помогают адаптировать задачи под изменения требований, что очень важно в проекте с высокой динамикой и множеством внешних интеграций.
3. **Минимизация рисков и раннее тестирование**: Каждая итерация заканчивается созданием работающего продукта, что позволяет своевременно обнаруживать и устранять ошибки, снижая затраты на исправление на более поздних стадиях.

2.2. Методология разработки программного обеспечения

Для реализации проекта «Билеты онлайн» была выбрана методология Scrum. Scrum идеально сочетается с agile-подходом, предоставляя структуру для организации командной работы, регулярного обновления функционала и взаимодействия с заказчиком. Команда разработки делит процесс на короткие спринты (обычно 2-4 недели), каждый из которых завершается демонстрацией готового инкремента системы. Это позволяет заказчику тестировать готовую часть продукта, а разработчикам — своевременно вносить изменения и улучшения.

Основные преимущества методологии Scrum:

1. **Прозрачность и открытость**: Каждый член команды знает свои задачи и цели, что улучшает координацию и ускоряет процесс разработки.

2. **Постоянное улучшение продукта**: За счёт регулярных демонстраций и тесного взаимодействия с заказчиком можно быстро выявлять проблемы и вносить корректировки, что снижает риски накопления ошибок и увеличивает удовлетворённость пользователей.

2.3. Сравнение с другими моделями жизненного цикла и методологиями разработки ПО

2.3.1. Модели жизненного цикла ПО

Водопадная модель (Waterfall)

Основные недостатки для проекта «Билеты онлайн»:

1. **Низкая гибкость**: Водопадная модель не позволяет вносить изменения в требования после завершения определенных этапов, что делает ее неподходящей для проекта с изменяющимися условиями и требованиями.
2. **Длительные циклы разработки**: Водопадная модель не предусматривает возможности частичной сдачи продукта на ранних этапах, что увеличивает риски несоответствия ожиданиям заказчика.

V-модель (V-Model)

Основные недостатки для проекта «Билеты онлайн»:

1. **Ограниченная гибкость**: V-модель предусматривает строгую последовательность этапов разработки и тестирования, что затрудняет внедрение изменений в требования на поздних стадиях проекта.
2. **Затрудненное управление изменениями**: Изменения в требованиях приводят к необходимости повторного прохождения всех стадий разработки, что увеличивает время и затраты.

Спиральная модель

Основные недостатки для проекта «Билеты онлайн»:

1. **Сложность управления:** Спиральная модель требует сложного управления и постоянного анализа рисков, что может затянуть процесс разработки, особенно при недостатке ресурсов.
2. **Высокие затраты:** Длительный процесс повторного анализа и оценки рисков на каждом витке спирали может привести к увеличению затрат и времени на проект, что не всегда оправдано.

2.3.2. Методологии разработки ПО

Kanban

Основные недостатки для проекта «Билеты онлайн»:

1. **Отсутствие чётких временных рамок:** Kanban не задает жестких временных границ для выполнения задач, что может привести к затягиванию сроков выполнения критически важных задач.
2. **Менее структурированный процесс:** По сравнению со Scrum, Kanban может быть менее эффективен для управления большими командами и сложными проектами.

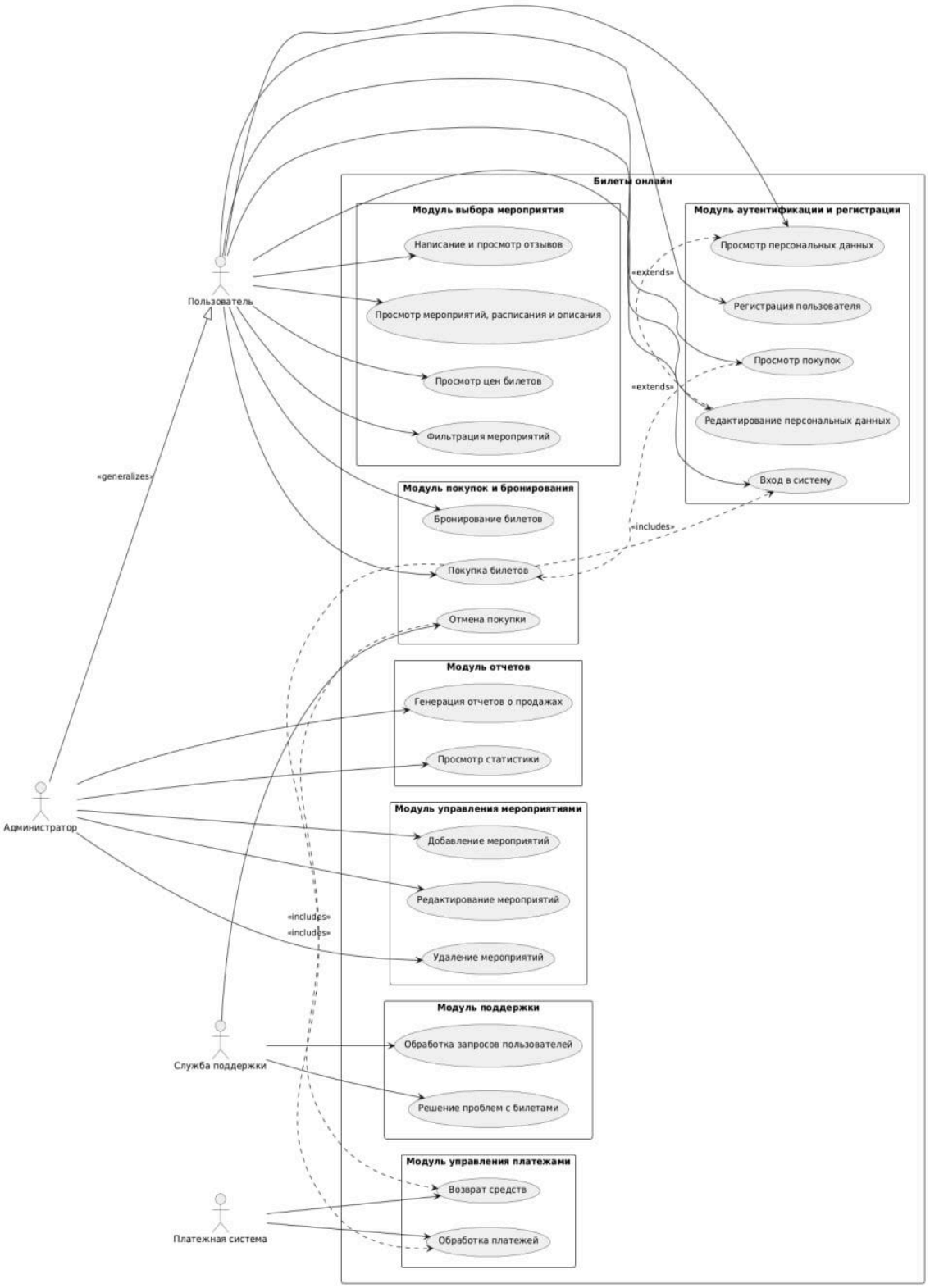
DevOps
Основные недостатки для проекта «Билеты онлайн»:

1. **Сложности внедрения:** DevOps требует глубокой интеграции процессов разработки и эксплуатации, что может быть сложно реализовать в команде с ограниченными ресурсами.
2. **Менее структурированный процесс:** По сравнению со Scrum, Kanban может быть менее эффективен для управления большими командами и сложными проектами.

XP (Extreme Programming)
Основные недостатки для проекта «Билеты онлайн»:

1. **Требование к высокой дисциплине:** XP требует строгого следования методологии и постоянного рефакторинга кода, что может быть сложно обеспечить в условиях динамично меняющихся требований.
2. **Сложность для больших команд:** XP изначально ориентирован на небольшие команды, что может усложнить его применение в проектах с более широким масштабом и множеством зависимостей.

3. Диаграмма прецедентов



1. Описание к диаграмме прецедентов.

На данной UML-диаграмме прецедентов(use case diagram) изображена структура системы онлайн-бронирования и продажи билетов “Билеты онлайн”, разделенная на несколько модулей с описанием связей между акторами и прецедентами. Вот описание диаграммы:

1.1. Акторы:

- **Пользователь** — человек, который использует систему для регистрации, покупки билетов, просмотра мероприятий и личных данных.
- **Администратор** — человек, который управляет мероприятиями и отчетами в системе. Является специализированной версией пользователя и наследует его возможности, но также имеет дополнительные права.
- **Служба поддержки** — сотрудники, помогающие пользователям в решении проблем с билетами и покупками.
- **Платежная система** — внешняя система, обрабатывающая платежи и возвраты средств.

1.2. Модули и прецеденты:

1. **Модуль аутентификации и регистрации:**
    - **Регистрация пользователя:** Пользователь регистрируется в системе.
    - **Вход в систему:** Пользователь выполняет вход в систему.
    - **Просмотр персональных данных:** Пользователь может просматривать свои личные данные в личном кабинете.
    - **Редактирование персональных данных:** Пользователь может редактировать свои данные.
        - Связь `<<extends>>`: Прецедент **"Редактирование персональных данных"** расширяет **"Просмотр персональных данных"**, так как редактирование требует предварительного просмотра данных.
    - **Просмотр покупок:** Пользователь может просматривать свои покупки.
        - Связь `<<extends>>`: Прецедент **"Просмотр покупок"** дополняет **"Покупка билетов"**, поскольку пользователь может просмотреть свои покупки после их совершения.
2. **Модуль выбора мероприятия:**
    - **Написание и просмотр отзывов:** Пользователь может оставить отзыв о мероприятии и просмотреть отзывы других пользователей.
    - **Просмотр мероприятий, расписания и описания:** Пользователь может просматривать расписание мероприятий и их описание.
    - **Просмотр цен билетов:** Пользователь может просмотреть цены на билеты.

- **Фильтрация мероприятий:** Пользователь может фильтровать мероприятия по различным параметрам.

3. **Модуль покупок и бронирования:**
   - **Покупка билетов:** Пользователь покупает билеты на мероприятие.
     - Связь `<<includes>>`: Прецедент **"Покупка билетов"** включает **"Вход в систему"**, так как для совершения покупки пользователь должен сначала войти в систему.
     - Связь `<<includes>>`: Прецедент **"Покупка билетов"** включает **"Обработку платежей"**, поскольку покупка билетов требует оплаты.
   - **Бронирование билетов:** Пользователь может забронировать билеты вместо их покупки.
   - **Отмена покупки:** Служба поддержки может отменить покупку билетов.
     - Связь `<<includes>>`: Прецедент **"Отмена покупки"** включает **"Возврат средств"**, так как возврат средств — это неотъемлемая часть процесса отмены покупки.
4. **Модуль управления платежами:**
   - **Обработка платежей:** Платежная система обрабатывает платежи за билеты.
   - **Возврат средств:** Платежная система может инициировать возврат средств, если покупка отменена.
5. **Модуль отчетов:**
   - **Генерация отчетов о продажах:** Администратор может генерировать отчеты по продажам билетов.
   - **Просмотр статистики:** Администратор может просматривать статистику продаж и активности пользователей.
6. **Модуль управления мероприятиями:**
   - **Добавление мероприятий:** Администратор добавляет новые мероприятия в систему.
   - **Редактирование мероприятий:** Администратор может редактировать информацию о мероприятиях.
   - **Удаление мероприятий:** Администратор удаляет мероприятия из системы.
7. **Модуль поддержки:**
   - **Обработка запросов пользователей:** Служба поддержки обрабатывает запросы пользователей.
   - **Решение проблем с билетами:** Служба поддержки решает проблемы, связанные с покупкой или бронированием билетов.

1.3. Связи:

- `<<extends>>`:

    - (Редактирование персональных данных) `<<extends>>` (Просмотр персональных данных): Пользователь может редактировать свои данные только после их просмотра.
    - (Просмотр покупок) `<<extends>>` (Покупка билетов): После совершения покупки пользователь может просмотреть свои билеты.
- `<<includes>>`:
    - (Покупка билетов) `<<includes>>` (Вход в систему): Покупка билетов требует предварительной аутентификации пользователя.
    - (Покупка билетов) `<<includes>>` (Обработка платежей): Обработка платежей является обязательным шагом при покупке билетов.
    - (Отмена покупки) `<<includes>>` (Возврат средств): Возврат средств происходит автоматически при отмене покупки.

1.4. Обобщение:

- (Администратор) `<<generalizes>>` (Пользователь): Администратор является специализированным пользователем и наследует все его функции, но имеет дополнительные права, такие как управление мероприятиями и отчетами.

2. Описание нормальных и аномальных сценариев

Нормальный сценарий №1
Выбор мероприятия:
1. Пользователь вводит параметры поиска (жанр, дата)
2. Система отображает список доступных мероприятий
3. Пользователь просматривает расписание, описание и цены билетов.
4. Пользователь выбирает мероприятие для дальнейших действий (покупка или бронирование)

Аномальный сценарий №1
Ошибка при выборе мероприятия:
1. Пользователь вводит параметры поиска (жанр, дата)
2. Система отправляет запрос, но возникает ошибка (проблема с подключением к базе данных)
3. Система сообщает пользователю об ошибке и предлагает попробовать позже или изменить фильтры.
4. Пользователь обновляет параметры фильтрации или ждет восстановления системы.

Нормальный сценарий №2

Покупка билета:
1. Пользователь выбирает мероприятие и место.
2. Пользователь нажимает кнопку “Купить билет”
3. Система предлагает ввести данные для оплаты или выбрать сохраненные данные.
4. Пользователь вводит платежные данные или выбирает уже сохраненные данные.
5. Пользователь подтверждает покупку.
6. Система отправляет подтверждение покупки и билет на электронную почту пользователя.

Аномальный сценарий №2
Ошибка при оплате билета
1. Пользователь выбирает мероприятие, место и пытается оплатить билет.
2. Система отправляет запрос в платежную систему, но происходит сбой.
3. Система уведомляет пользователя об ошибке и предлагает выбрать другой способ оплаты или попробовать позже.
4. Пользователь повторяет попытку или выбирает альтернативный метод оплаты.

Нормальный сценарий №3
Возврат средств:
1. Пользователь запрашивает возврат средств за купленный билет
2. Система проверяет допустимость возврата.
3. Если возврат возможен, система инициирует процесс.
4. Система уведомляет пользователя о возврате средств и отправляет подтверждение на почту.
5. Средства возвращаются на счет пользователя.

Аномальный сценарий №3
Отмена покупки вне срока возврата:
1. Пользователь пытается отменить покупку билета.
2. Система проверяет сроки возврата и уведомляет пользователя, что возврат уже невозможен.
3. Пользователь может выбрать отмену без возврата средств или отказаться от отмены.
4. Система отправляет уведомление о статусе отмены.

3. Почему были выбраны именно эти use-кейсы для описания сценария?

Основываясь на роли в диаграмме и значимости для функциональности системы, были выбраны данные use-кейсы:

1. **Выбор мероприятия**
Этот модуль - основное место взаимодействия пользователя с системой, где пользователь определяет, какое мероприятие он хочет посетить. Фильтрация мероприятий, просмотр расписания и цен это ключевые функции для того, чтобы пользователь мог сделать выбор. Нормальные и аномальный сценарии выбора мероприятия необходимы для определения, насколько удобно пользователю найти подходящее событие.

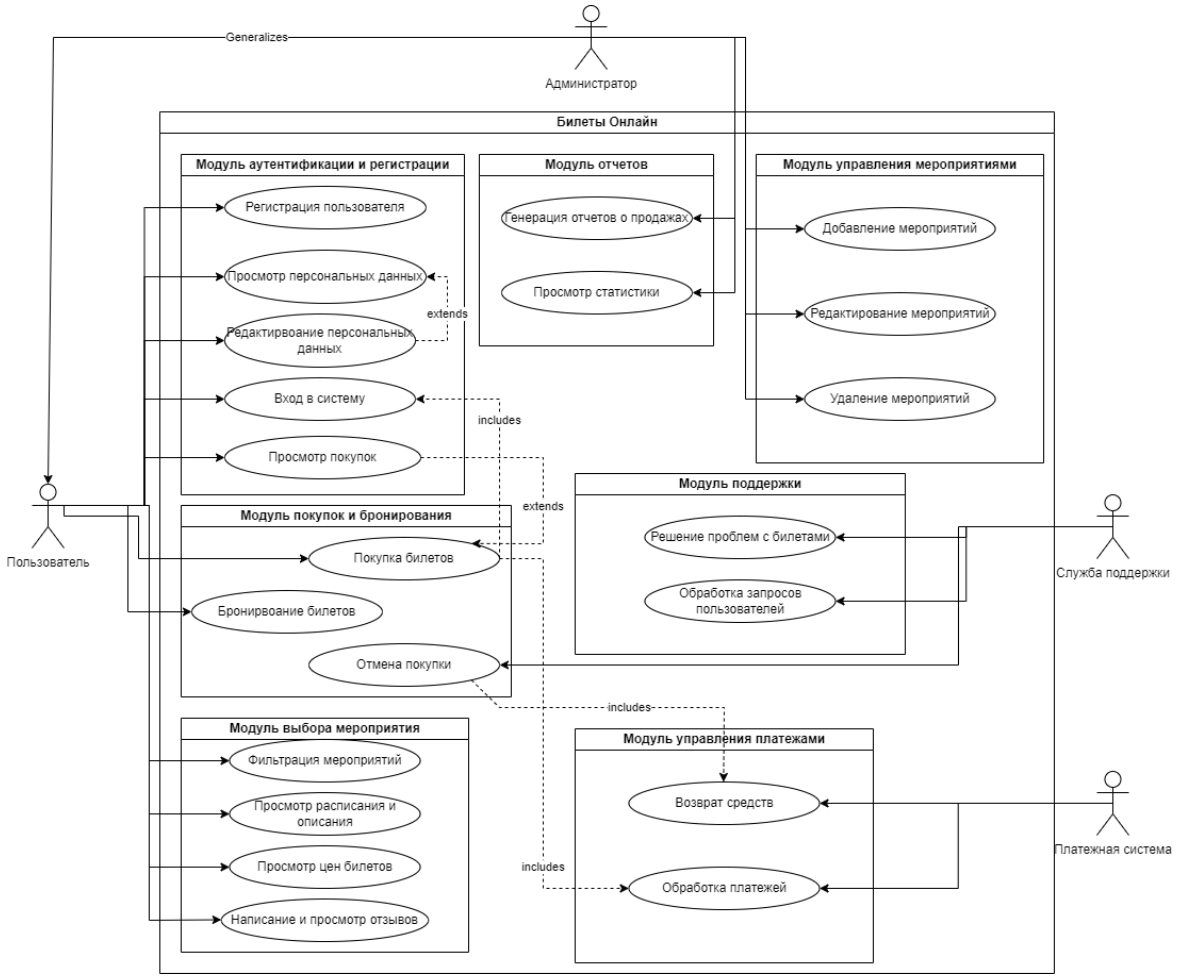
2. **Покупка билета**
Процесс покупки билета — это один из самых важных и частых действий, который выполняет пользователь. Он включает такие шаги, такие как выбор места, ввод платежных данных и подтверждение покупки или бронирования. От того, насколько этот процесс работает гладко или с ошибками, зависит удовлетворенность пользователя. Ошибки при покупке или проблемы с оплатой могу привести к отказу от использования от системы, что делает данные сценарии важными для анализа.

3. **Возврат средств**
Возврат средств — это важная часть клиентского сервиса, которая напрямую влияет на удовлетворенность пользователей. Если пользователь не может посетить мероприятие или отменяет покупку, возврат должен происходить корректно и в срок. Аномальные сценарии возврата (например, отмена после срока возврата) — частая ситуация, которая требует четкого объяснения пользователю.

Общие причины выбора данных use-кейсов:
Ключевая функциональность: Все выбранные кейсы относятся к основным функциям системы — выбор мероприятий, покупка билетов, возврат средств. Это функциональные блоки, от которых зависит успешное использование системы.

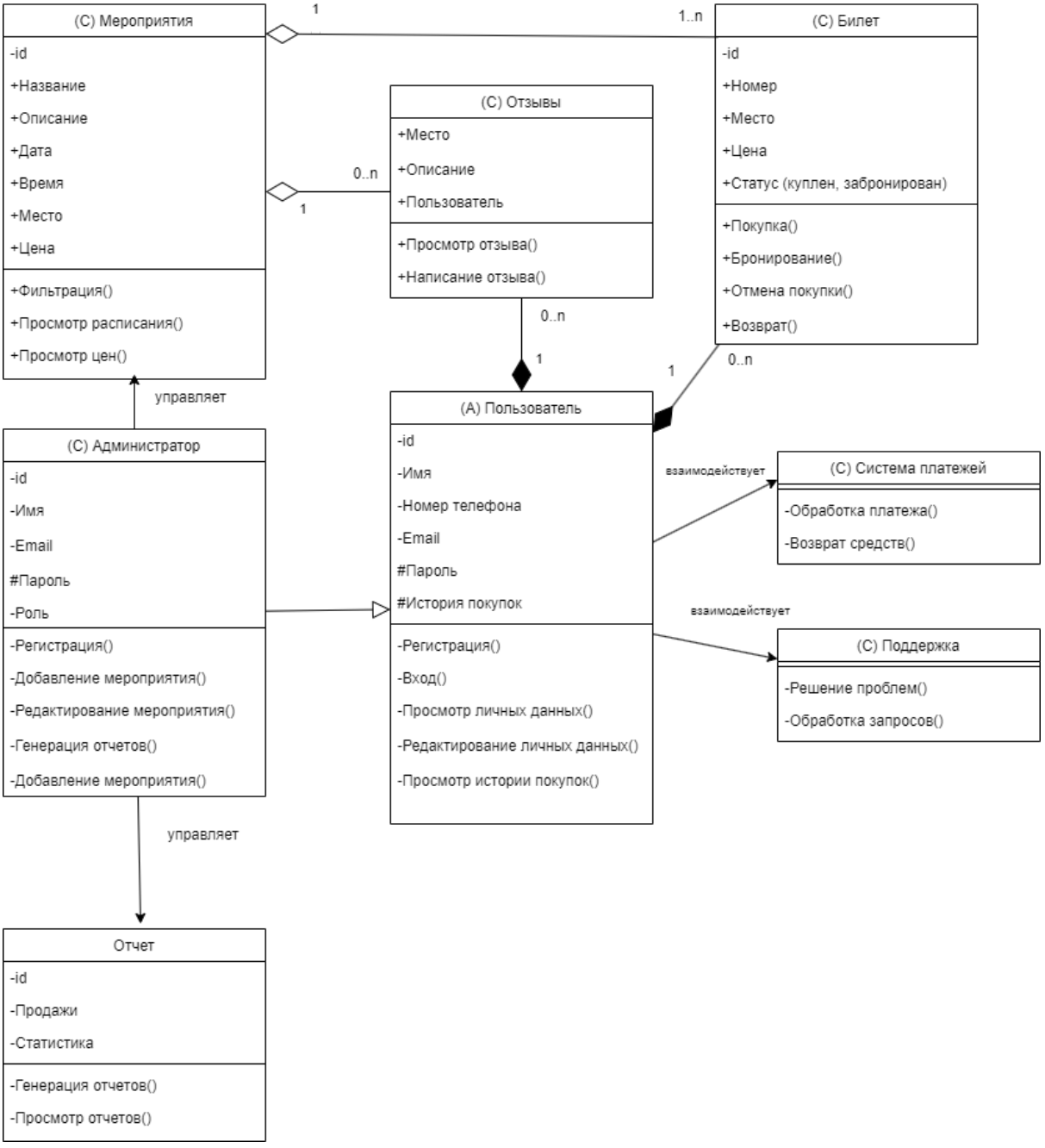
Важность для пользователя: Кейсы, связанные с покупкой билетов и возвратом, напрямую влияют на пользовательский опыт. Ошибки или сбои в этих процессах приведут к потере лояльности клиентов.

Диаграмма прецедентов сделанная в среде draw io



4. Диаграмма классов

1. Первичное проектирование. Выявление первичных классов.



1.1. Первичное проектирование

Классы:

1. Пользователь (Абстрактный класс)
    - Атрибуты:
        - id: int - уникальный идентификатор пользователя
        - Имя: str - имя пользователя

- Номер телефона: str - телефонный номер пользователя
- Email: str - адрес электронной почты пользователя
- Пароль: str - пароль для доступа к системе
- История покупок: list - список билетов

- Методы:
  - Регистрация(имя: str, номер_телефона: str, email: str, пароль: str) - метод для регистрации нового пользователя
  - Вход(email: str, пароль: str) — метод для входа пользователя в систему
  - Просмотр_личных_данных(): dict — возвращает личные данные пользователя (имя, номер телефона, email)
  - Редактирование_личных_данных(имя: str, номер_телефона: str, email: str) — метод для редактирования личных данных
  - Просмотр_истории_покупок(): list — возвращает список всех покупок пользователя

2. Администратор (Наследует Пользователя)
   - Атрибуты:
     -id: int — уникальный идентификатор администратора
     -Имя: str — имя администратора
     -Email: str — адрес электронной почты администратора
     -Пароль: str - пароль для доступа к системе
     -Роль: str — роль администратора
   - Методы:
     -Регистрация(имя: str, email: str, пароль: str) — метод регистрации администратора
     -Вход(email: str, пароль: str) — метод для входа администратора в систему
     -Просмотр_личных_данных(): dict — возвращает личные данные администратора
     -Добавление_мероприятия(название: str, дата: datetime, место: str) — метод для добавления нового мероприятия в систему
     -Редактирование_мероприятия(id: int, название: str, дата: datetime, место: str) — редактирование информации о мероприятии
     -Удаление_мероприятия(id: int) — метод для удаления мероприятия по его идентификатору
     -Генерация_отчетов(): list — метод для создания отчетов о продажах и статистике
3. Мероприятия
   - Атрибуты:
     -id: int — уникальный идентификатор мероприятия
     -Название: str — название мероприятия
     -Описание: str — краткое описание мероприятия
     -Дата: datetime — дата проведения мероприятия
     -Время: time — время начала мероприятия
     -Место: str — место проведения мероприятия

-Цена: float — цена билета на мероприятие

- Методы:
  -Фильтрация(параметры: dict): list — фильтрует мероприятия по заданным параметрам
  -Просмотр_расписания(): list — возвращает расписание всех мероприятий
  -Просмотр_цен(): list — возвращает список цен на билеты для каждого мероприятия

4. Отчет
   - Атрибуты:
     -id: int — уникальный идентификатор отчета
     -Продажи: float — сумма продаж за период
     -Статистика: dict — статистическая информация
   - Методы:
     -Генерация_отчетов(параметры: dict): list — генерация отчета на основе заданных параметров
     -Просмотр_отчетов(): list — просмотр ранее созданных отчетов
5. Отзывы
   - Атрибуты:
     -Место: str — место, где проходило мероприятие
     -Описание: str — текст отзыва.
     -Пользователь: Пользователь — объект пользователя, оставившего отзыв
   - Методы:
     -Просмотр_отзыва(id: int): dict — возвращает информацию о конкретном отзыве
     -Написание_отзыва(место: str, описание: str) — метод для добавления отзыва
6. Билет
   - Атрибуты:
     -id: int — уникальный идентификатор билета
     -Номер: str — номер билета
     -Место: str — место, указанное на билете (ряд и место)
     -Цена: float — стоимость билета
     -Статус: str — статус билета (куплен, забронирован, отменен)
   - Методы:
     -Покупка(пользователь: Пользователь, мероприятие: Мероприятие) — покупка билета на указанное мероприятие
     -Бронирование(пользователь: Пользователь, мероприятие: Мероприятие) — бронирование билета на мероприятие
     -Отмена_покупки(id: int) — отмена покупки билета
     -Возврат(id: int) — возврат билета и средств
7. Система платежей
   - Методы:

-Обработка_платежа(пользователь: Пользователь, сумма: float) — метод для обработки платежа за билет
-Возврат_средств(пользователь: Пользователь, id: int) — метод для возврата средств за отмененные билеты

8. Поддержка:
    - Методы:
      -Решение_проблем(запрос: str): str — метод для решения проблем, связанных с покупкой или бронированием билетов
      -Обработка_запросов(пользователь: Пользователь, запрос: str) — метод для обработки запросов пользователей

1.2. Полиморфные и абстрактные методы:

Абстрактные методы в классе Пользователь:
входВСистему():
Описание: Этот метод должен быть реализован в классах-наследниках, так как логика входа может различаться. Например, администраторы могут требовать дополнительные уровни аутентификации.

регистрация():
Описание: Этот метод будет определять процесс регистрации пользователей. Абстрактный метод позволит каждому классу-наследнику (например, администратору и обычному пользователю) реализовать свою специфику регистрации.

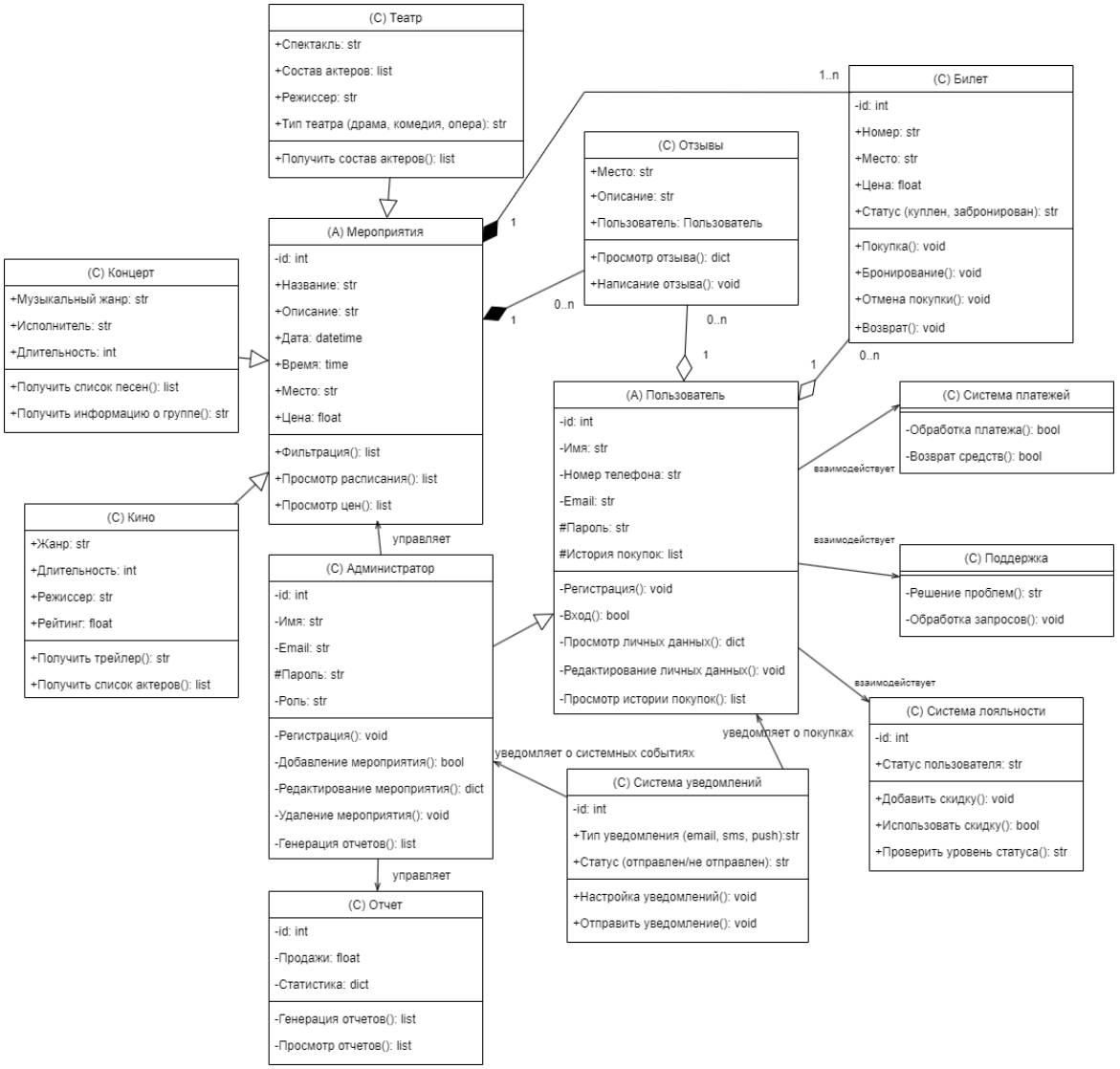
просмотрИсторииПокупок():
Описание: Этот метод должен быть реализован в каждом классе-наследнике, чтобы обеспечить соответствующий уровень доступа к истории покупок. Например, администратор может видеть историю всех пользователей, а обычный пользователь — только свою.

Полиморфные методы в классе Пользователь:
регистрация():
Описание: Полиморфный метод, который будет реализован по-разному в классах Администратор и Обычный пользователь.
Регистрация администратора: Может включать дополнительные шаги, такие как подтверждение роли и предоставление прав доступа к определенным функциям системы.
Регистрация обычного пользователя: Стандартная процедура регистрации, которая может включать ввод имени, номера телефона, электронной почты и пароля без дополнительных проверок прав.

1.3. Связи:

Мероприятие - Отзывы: композиция, отзыв не может существовать без мероприятия. Если мероприятие удаляется, отзывы тоже.
Мероприятие - Билет: композиция, билеты зависят от мероприятия. Удаление мероприятия удаляет все билеты.
Администратор - Мероприятие: ассоциация, администратор управляет мероприятиями, но мероприятия могут существовать независимо.
Администратор - Отчет: ассоциация, администратор создает отчеты, которые могут существовать независимо от него.
Администратор - Пользователь: наследование, администратор обладает всеми возможностями пользователя и расширяет его функционал дополнительными правами (например, управление мероприятиями)
Пользователь - Билет: агрегация, пользователь покупает билеты, но они могут существовать без него (например, после отмены).
Пользователь - Отзывы: агрегация, пользователь создает и просматривает, но они могут существовать независимо от него.
Пользователь - Система платежей: ассоциация, пользователь взаимодействует с системой платежей для оплаты, но они могут существовать независимо.
Пользователь - Поддержка: ассоциация, пользователь обращается в поддержку, но обе стороны существуют независимо.
Билет - Система платежей: ассоциация, билеты связаны с системой для обработки платежей, но могут существовать независимо.
Система платежей - Поддержка: ассоциация, система платежей и поддержка взаимодействуют по вопросам возврата, но не зависят друг от друга.

2. Детальная диаграмма классов



Классы:

1. Пользователь (Абстрактный класс)
    - Атрибуты:
      - id: int - уникальный идентификатор пользователя
      - Имя: str - имя пользователя
      - Номер телефона: str - телефонный номер пользователя
      - Email: str - адрес электронной почты пользователя
      - Пароль: str - пароль для доступа к системе
      - История покупок: list - список билетов
    - Методы:
      - Регистрация(имя: str, номер_телефона: str, email: str, пароль: str): void - метод для регистрации нового пользователя
      - Вход(email: str, пароль: str): bool — метод для входа пользователя в систему

- Просмотр_личных_данных(): dict — возвращает личные данные пользователя (имя, номер телефона, email)
- Редактирование_личных_данных(имя: str, номер_телефона: str, email: str): void — метод для редактирования личных данных
- Просмотр_истории_покупок(): list — возвращает список всех покупок пользователя

2. Администратор (Наследует Пользователя)
    - Атрибуты:
      -id: int — уникальный идентификатор администратора
      -Имя: str — имя администратора
      -Email: str — адрес электронной почты администратора
      -Пароль: str - пароль для доступа к системе
      -Роль: str — роль администратора
    - Методы:
      -Регистрация(имя: str, email: str, пароль: str): void — метод регистрации администратора
      -Вход(email: str, пароль: str): bool — метод для входа администратора в систему
      -Просмотр_личных_данных(): dict — возвращает личные данные администратора
      -Добавление_мероприятия(название: str, дата: datetime, место: str): void — метод для добавления нового мероприятия в систему
      -Редактирование_мероприятия(id: int, название: str, дата: datetime, место: str): void — редактирование информации о мероприятии
      -Удаление_мероприятия(id: int): void — метод для удаления мероприятия по его идентификатору
      -Генерация_отчетов(): list — метод для создания отчетов о продажах и статистике
3. Мероприятия
    - Атрибуты:
      -id: int — уникальный идентификатор мероприятия
      -Название: str — название мероприятия
      -Описание: str — краткое описание мероприятия
      -Дата: datetime — дата проведения мероприятия
      -Время: time — время начала мероприятия
      -Место: str — место проведения мероприятия
      -Цена: float — цена билета на мероприятие
    - Методы:
      -Фильтрация(параметры: dict): list — фильтрует мероприятия по заданным параметрам
      -Просмотр_расписания(): list — возвращает расписание всех мероприятий

-Просмотр_цен(): list — возвращает список цен на билеты для каждого мероприятия

4. Отчет
    - Атрибуты:
      -id: int — уникальный идентификатор отчета
      -Продажи: float — сумма продаж за период
      -Статистика: dict — статистическая информация
    - Методы:
      -Генерация_отчетов(параметры: dict): list — генерация отчета на основе заданных параметров
      -Просмотр_отчетов(): list — просмотр ранее созданных отчетов
5. Отзывы
    - Атрибуты:
      -Место: str — место, где проходило мероприятие
      -Описание: str — текст отзыва.
      -Пользователь: Пользователь — объект пользователя, оставившего отзыв
    - Методы:
      -Просмотр_отзыва(id: int): dict — возвращает информацию о конкретном отзыве
      -Написание_отзыва(место: str, описание: str): void — метод для добавления отзыва
6. Билет
    - Атрибуты:
      -id: int — уникальный идентификатор билета
      -Номер: str — номер билета
      -Место: str — место, указанное на билете (ряд и место)
      -Цена: float — стоимость билета
      -Статус: str — статус билета (куплен, забронирован, отменен)
    - Методы:
      -Покупка(пользователь: Пользователь, мероприятие: Мероприятие): void — покупка билета на указанное мероприятие
      -Бронирование(пользователь: Пользователь, мероприятие: Мероприятие): void— бронирование билета на мероприятие
      -Отмена_покупки(id: int): void — отмена покупки билета
      -Возврат(id: int): void — возврат билета и средств
7. Система платежей
    - Методы:
      -Обработка_платежа(пользователь: Пользователь, сумма: float): bool — метод для обработки платежа за билет
      -Возврат_средств(пользователь: Пользователь, id: int): bool — метод для возврата средств за отмененные билеты
8. Система уведомлений
    - Атрибуты:
      -id: int — уникальный идентификатор уведомления

-Тип уведомления: str — тип уведомления (email, sms, push)
-Статус: str — статус уведомления (отправлено/не отправлено)

- Методы:
-Настройка_уведомлений(): void — настройка параметров уведомлений для пользователя
-Отправка_уведомлений(): void — отправка уведомления пользователю.

9. Система лояльности
    - Атрибуты:
    -id: int — уникальный идентификатор системы лояльности
    -Статус пользователя: str — уровень лояльности пользователя
    - Методы:
    -Добавить_скидку(сумма: float): void — добавляет скидку пользователю в зависимости от его статуса
    -Использовать_скидку(сумма: float): bool — применяет скидку при покупке
    -Проверить_уровень_статуса(): str — проверяет уровень статуса пользователя
10. Театр
    - Атрибуты:
    -Спектакль: str — название спектакля
    -Состав актеров: list — список актеров, участвующих в спектакле
    -Режиссер: str — имя режиссера
    -Тип спектакля: str — жанр спектакля (драма, комедия, опера)
    - Методы:
    -Получить_состав_актеров(): list — возвращает список актеров, участвующих в спектакле
11. Концерт
    - Атрибуты:
    - Музыкальный жанр: str — жанр концерта (поп, рок, джаз и т.д.)
    -Исполнитель: str — имя исполнителя или группы
    -Длительность: int — продолжительность концерта в минутах
    - Методы:
    -Получить_список_песен(): list — возвращает список песен, исполняемых на концерте
    -Получить_информацию_о_группе(): str — возвращает информацию о группе или исполнителе
12. Кино
    - Атрибуты:
    -Жанр: str — жанр фильма (драма, комедия, боевик и т.д.)
    -Длительность: int — продолжительность фильма в минутах
    -Режиссер: str — имя режиссера фильма
    -Рейтинг: float — рейтинг фильма

- Методы:
  -Получить_трейлер(): str — возвращает ссылку на трейлер фильма
  -Получить_список_актеров(): list — возвращает список актеров, участвующих в фильме

13. Поддержка:
    - Методы:
      -Решение_проблем(запрос: str): str — метод для решения проблем, связанных с покупкой или бронированием билетов
      -Обработка_запросов(пользователь: Пользователь, запрос: str): void — метод для обработки запросов пользователей

2.1. Зависимости и отношения между классами:

• **Мероприятие — композиция — Отзывы** (мероприятие содержит отзывы, и отзывы не могут существовать без мероприятия)
Мероприятие(1) --- (0…*) Отзывы
Отзывы удаляются вместе с мероприятием.

• **Мероприятие — композиция — Билет** (билеты привязаны к мероприятиям и не могут существовать без них)
Мероприятие(1) --- (0…*) Билет
Удаление мероприятия приводит к удалению всех связанных билетов.

• **Пользователь — агрегация — Отзывы** (пользователь создает отзывы, но они могут существовать независимо от пользователя)
Пользователь(1) --- (0…*) Отзывы
Отзывы остаются, даже если пользователь удаляется.

• **Пользователь — агрегация — Билет** (один пользователь может купить несколько билетов, но билеты могут существовать отдельно от пользователя)
Пользователь(1) --- (0…*) Билет
Билеты продолжают существовать независимо от пользователя.

• **Администратор — ассоциация — Мероприятия** (администратор управляет мероприятиями, но мероприятия могут существовать без администратора)
Администратор(1) --- (0…*) Мероприятие
Администратор может добавлять, редактировать и удалять мероприятия.

• **Администратор — ассоциация — Отчет** (администратор генерирует отчеты по мероприятиям, но отчеты могут существовать независимо от администратора)
Администратор(1) --- (0…*) Отчет
Отчеты создаются на основе данных о мероприятиях.

• **Пользователь — ассоциация — Система платежей** (пользователь взаимодействует с системой для оплаты билетов, но система существует независимо)
Пользователь(1) --- (1) Система платежей
Система обрабатывает платежи и возвраты средств для пользователя.

• **Пользователь — ассоциация — Поддержка** (пользователь взаимодействует с поддержкой для решения проблем, но поддержка и пользователь существуют независимо)
Пользователь(1) --- (0…1) Поддержка
Поддержка помогает решать проблемы, связанные с покупками и билетом.

• **Пользователь — ассоциация — Система лояльности** (пользователь взаимодействует с системой лояльности для начисления и использования баллов, но система и пользователь существуют независимо)
Система лояльности(1) --- (0…*) Пользователь
Баллы начисляются за действия пользователей в системе.

• **Система платежей — ассоциация — Билет**
Система платежей(1) --- (0…*) Билет
Платежи за билеты обрабатываются системой, но билеты могут существовать после их оплаты.

• **Пользователь — ассоциация — Система уведомлений** (пользователь получает уведомления о системных событиях, таких как покупка билетов)
Система уведомлений(1) --- (0…*) Пользователь
Система уведомлений уведомляет пользователей о событиях, связанных с их деятельностью.

• **Система уведомлений — ассоциация — Администратор**
Система уведомлений уведомляет администратора о системных событиях.
Система уведомлений --- (1) --- (0..*) Администратор

• **Администратор — наследование — Пользователь**
Администратор наследует атрибуты и методы пользователя, но имеет расширенные права.

• **Концерт — наследование — Мероприятие**
Концерт наследует от Мероприятие, но добавляет атрибуты, такие как музыкальный жанр, исполнитель, и методы для получения информации о группе и списке песен.

• **Театр — наследование — Мероприятие**
Театр наследует от Мероприятие, но добавляет атрибуты, такие как состав актеров, режиссер, и методы для получения информации о спектаклях и типе театра (драма, комедия, опера)

• Кино — наследование — Мероприятие
Кино наследует от Мероприятие, но добавляет атрибуты, такие как жанр фильма, длительность, рейтинг, и методы для получения трейлеров и информации об актерах.

5. Выбор архитектуры

Нами была выбрана микросервисная архитектура, так как данный подход будет наиболее эффективным для создания нашего ПО.
Преимущества:

Почему микросервисная архитектура:

1. **Масштабируемость**: Каждый компонент системы (например, модуль управления билетами, платежная система, система лояльности) может быть определен в отдельный микросервис, это позволит легко масштабировать наиболее нагруженные компоненты(например, обработка билетов на популярные мероприятия).
2. **Независимость компонентов**: Микросервисы позволяют разворачивать и обновлять отдельные компоненты системы без остановки всей платформы, иначе это может быть критическим фактором для системы с высокой нагрузкой, особенно во время массовых продаж билетов на популярные мероприятия.
3. **Гибкость разработки**: Разработку можно распределить между разными людьми или командами, чтобы каждая группа работала над своим сервисом. Это ускоряет процесс внедрения новых функций (например, добавление новых методов оплаты или интеграция с внешними системами).
4. **Интеграция с внешними системами**: Наш проект требует интеграции с платежными шлюзами и API для автоматического обновления информации о мероприятиях. Микросервисная архитектура позволит легко управлять такими интеграциями, делая их независимыми от других частей системы.
5. **Отказоустойчивость**: В случае сбоя одного микросервиса (например, модуля отзывов) остальные будут продолжать работать, что значительно повышает стабильность системы, что, в свою очередь, повышает прибыль.
6. **Поддержка DevOps и CI/CD**: Микросервисы легко интегрируются с практиками непрерывной интеграции и развертывания, что позволит быстрее выпускать обновления и исправления.

1. Сравнение относительно других архитектур

Архитектура	Масштабируемость и гибкость	Обслуживание и развитие	Сложность разработки (простота)	Риск сбоев (чем выше, тем лучше)	Совместимость	Сумма
Монолитная	3/10	4/10	8/10	5/10	7/10	27/50
Микросервисная	9/10	8/10	5/10	8/10	8/10	38/50
Бессерверная	7/10	8/10	6/10	7/10	6/10	34/50
Сервис-ориентированная	7/10	7/10	6/10	7/10	8/10	35/50
Управляемая событиями	6/10	6/10	5/10	8/10	7/10	32/50
Клиент-серверная	6/10	6/10	7/10	6/10	7/10	32/50

Обоснование использования элементов архитектуры

• Тип клиента

Тонкий клиент (Web и мобильное приложение)

Это решение позволяет пользователям легко получать доступ к системе с любого устройства через браузер или мобильное приложение, что улучшает удобство и делает обновления системы централизованными.

• Реляционная база данных (PostgreSQL, MySQL)

Реляционная база данных подходит для хранения данных о пользователях, мероприятиях и билетах. Она обеспечивает быстрый доступ к данным, что важно для удобного поиска и покупки билетов.

• Облачное хранилище

Облачное хранилище, например Yandex Object Storage, может быть хорошим решением, чтобы хранить данные и предоставлять доступ к ним(использование в рамках Yandex.Cloud). Этот сервис может быть использован для решения для разных задач, включая хранение файлов, бэкапов данных, статических ресурсов для веб-приложений и так далее.

• Платёжные шлюзы (ЮMoney, СБП)

Интеграция с платёжными системами позволит безопасно и быстро проводить транзакции, предлагая пользователям различные методы оплаты и соблюдая стандарты безопасности, такие как PCI DSS.

• Балансировщики нагрузки

Балансировщики нагрузки помогают равномерно распределить входящий трафик

между серверами, что предотвращает перегрузку системы при пиковых продажах билетов.

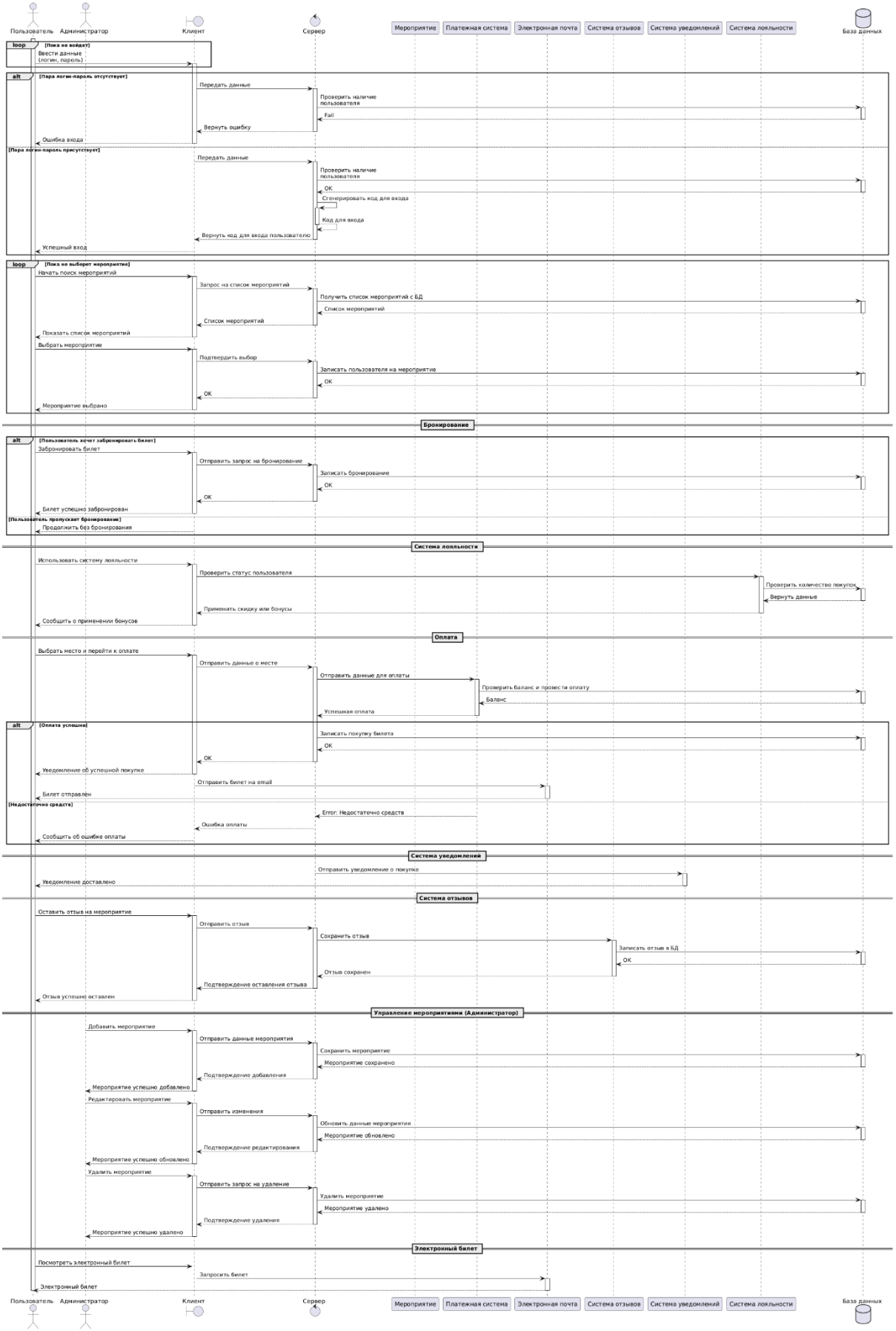
• Система кэширования
Системы кэширования могут использоваться для кэширования часто запрашиваемых данных, таких как списки мероприятий, профили пользователей и истории покупок. Это позволит значительно снизить нагрузку на базу данных и ускорить отклик системы для пользователей, особенно при массовых продажах билетов.

• CDN
Использование CDN помогает ускорить загрузку статических ресурсов(например, изображений и билетов) и обеспечивает равномерное распределение нагрузки. Это особенно важно для глобального охвата пользователей, которые могут находиться в разных странах, где время загрузки может отличаться.

• Мониторинг и логирование
Для эффективного мониторинга и быстрого реагирования на инциденты система должна использовать инструменты мониторинга, например, Prometheus для наблюдения за метриками производительности, и ELK Stack для логирования. Эти инструменты помогут отслеживать состояние микросервисов, анализировать возможные сбои и вовремя принимать меры для обеспечения бесперебойной работы системы.

6. Диаграмма последовательности

Код диаграммы последовательности:
https://docs.google.com/document/d/1TDu_Kf9dHCXVk62vhrkQ22UEnY0nUtUIighsSqofqOI/edit?usp=sharing

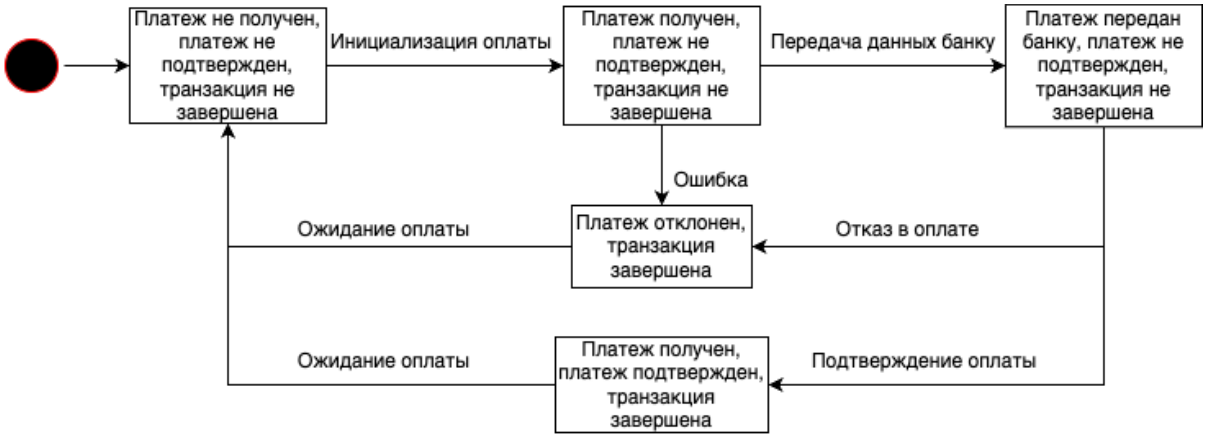

Пользователь
Администратор
Клиент
Сервер
Мероприятие
Платежная система
Электронная почта
Система отзывов
Система уведомлений
Система лояльности
База данных
loop
[Пока не войдет]
Ввести данные (логин, пароль)
alt
[Пара логин-пароль отсутствует]
Передать данные
Проверить наличие пользователя
Fail
Вернуть ошибку
Ошибка входа
[Пара логин-пароль присутствует]
Передать данные
Проверить наличие пользователя
OK
Сгенерировать код для входа
Код для входа
Вернуть код для входа пользователю
Успешный вход
loop
[Пока не выберет мероприятие]
Начать поиск мероприятий
Запрос на список мероприятий
Получить список мероприятий с БД
Список мероприятий
Список мероприятий
Показать список мероприятий
Выбрать мероприятие
Подтвердить выбор
Записать пользователя на мероприятие
OK
OK
Мероприятие выбрано
Бронирование
alt
[Пользователь хочет забронировать билет]
Забронировать билет
Отправить запрос на бронирование
Записать бронирование
OK
OK
Билет успешно забронирован
[Пользователь пропускает бронирование]
Продолжить без бронирования
Система лояльности
Использовать систему лояльности
Проверить статус пользователя
Проверить количество покупок
Вернуть данные
Применить скидку или бонусы
Сообщить о применении бонусов
Оплата
Выбрать место и перейти к оплате
Отправить данные о месте
Отправить данные для оплаты
Проверить баланс и провести оплату
Баланс
Успешная оплата
alt
[Оплата успешна]
Записать покупку билета
OK
OK
Уведомление об успешной покупке
Отправить билет на email
Билет отправлен
[Недостаточно средств]
Error: Недостаточно средств
Ошибка оплаты
Сообщить об ошибке оплаты
Система уведомлений
Отправить уведомление о покупке
Уведомление доставлено
Система отзывов
Оставить отзыв на мероприятие
Отправить отзыв
Сохранить отзыв
Записать отзыв в БД
OK
Отзыв сохранен
Подтверждение оставления отзыва
Отзыв успешно оставлен
Управление мероприятиями (Администратор)
Добавить мероприятие
Отправить данные мероприятия
Сохранить мероприятие
Мероприятие сохранено
Подтверждение добавления
Мероприятие успешно добавлено
Редактировать мероприятие
Отправить изменения
Обновить данные мероприятия
Мероприятие обновлено
Подтверждение редактирования
Мероприятие успешно обновлено
Удалить мероприятие
Отправить запрос на удаление
Удалить мероприятие
Мероприятие удалено
Подтверждение удаления
Мероприятие успешно удалено
Электронный билет
Посмотреть электронный билет
Запросить билет
Электронный билет

7. Диаграмма состояний

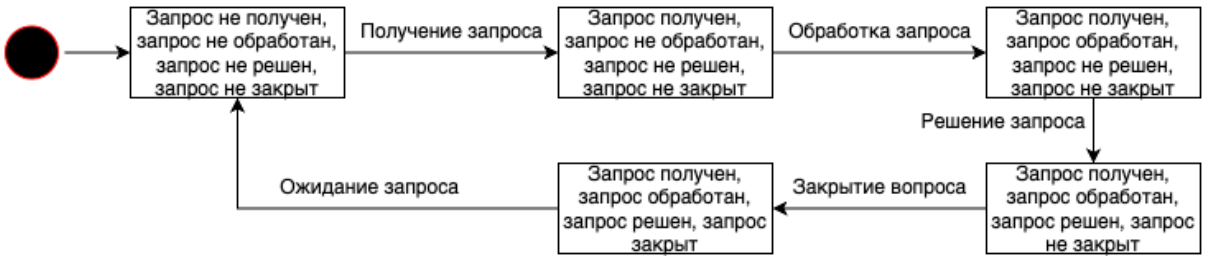
1. Диаграмма состояний пользователя

2. Диаграмма состояний администратора

3. Диаграмма состояний платежной системы



4. Диаграмма состояний службы поддержки



8. Матрица трассировки

Требование	Тест-кейс	Функциональное требование	Нефункциональное требование	Бизнес-требование
Регистрация и личный кабинет	ТК №1	Возможность регистрации пользователей через email, телефон или социальные сети.	Скорость отклика системы, защита персональных данных	Удобство доступа пользователей, повышение безопасности данных
	ТК №2	Восстановление пароля через email и телефон.	Безопасность данных, простота процесса восстановления	Увеличение удобства для пользователей
	ТК №3	Вход в личный кабинет, изменение данных.	Скорость отклика, масштабируемость	Поддержка актуальности данных пользователей
Поиск мероприятий и фильтрация	ТК №4	Поиск мероприятий по дате, месту, цене, типу.	Масштабируемость системы, оптимизация производительности	Повышение удобства использования, увеличение продаж
	ТК №5	Сортировка мероприятий по популярности и новизне.	Скорость отклика, оптимизация кода	Привлечение пользователей к популярным мероприятиям
Оформление покупки	ТК №6	Выбор места в зале и оплата билетов через несколько методов оплаты.	Безопасность данных при транзакциях, поддержка разных методов оплаты	Увеличение конверсии покупок и снижение отказов пользователей
	ТК №7	Получение электронного билета после успешной оплаты.	Надежность системы, своевременная отправка уведомлений	Увеличение доверия к системе и минимизация жалоб
Система лояльности для постоянных клиентов	ТК №8	Управление скидками и бонусами для постоянных клиентов.	Масштабируемость, гибкость системы	Увеличение лояльности клиентов и повторных покупок
Поддержка пользователей	ТК №9	Возможность чата с поддержкой, отправка заявок на возврат	Доступность системы поддержки 24/7	Улучшение клиентского опыта, снижение числа обращений в поддержку

		билетов и решение проблем через чат.		
Уведомления	ТК №10	Отправка push-уведомлений о новых мероприятиях и скидках пользователям.	Скорость доставки уведомлений, масштабируемость	Повышение информированности пользователей, увеличение вовлеченности
Управление мероприятиями (Администратор)	ТК №11	Создание нового мероприятия, редактирование информации, удаление мероприятий.	Масштабируемость системы, производительность	Управление активностями для увеличения продаж и привлечения клиентов
	ТК №12	Добавление и удаление категорий мероприятий.	Гибкость системы, поддержка обновлений	Легкость управления предложениями на платформе
Управление пользователями (Администратор)	ТК №13	Просмотр списка пользователей, изменение статуса (например, постоянный клиент), блокировка пользователей.	Безопасность данных, защита прав доступа	Обеспечение безопасности и эффективности управления пользователями
Обработка запросов (Администратор)	ТК №14	Обработка заявок на возврат билетов и модерация отзывов пользователей.	Надежность системы, минимизация сбоев	Повышение уровня обслуживания клиентов
Отчеты и аналитика (Администратор)	ТК №15	Формирование отчетов о продажах, активности пользователей.	Производительность системы, поддержка аналитики	Оптимизация предложений для пользователей на основе данных
Управление безопасностью	ТК №16	Проверка безопасности данных, соответствие стандартам (например, PCI DSS).	Безопасность данных, поддержка сертификаций	Уверенность в надежности системы и защита данных пользователей

Тест-кейсы, разработанные к системе

Требование №1. Регистрация и личный кабинет

Тест-кейс №1: Регистрация пользователей через email, телефон или социальные сети

Описание	Регистрация пользователя через email, телефон или социальные сети
Предусловие	Пользователь не зарегистрирован
Шаги	1. Пользователь переходит на страницу регистрации. 2. Вводит свои данные для регистрации (email, телефон, или выбирает социальную сеть). 3. Подтверждает регистрацию.
Ожидаемый результат	Пользователь успешно зарегистрирован, отправлено подтверждение на email/телефон.

Тест-кейс №2: Восстановление пароля через email и телефон

Описание	Восстановление пароля через email или телефон
Предусловие	Пользователь забыл пароль
Шаги	1. Пользователь переходит на страницу восстановления пароля. 2. Вводит email или телефон. 3. Подтверждает отправку данных. 4. Получает код/ссылку для восстановления.
Ожидаемый результат	Пользователь успешно восстановил пароль.

Тест-кейс №3: Вход в личный кабинет и изменение данных

Описание	Вход в личный кабинет и изменение данных
Предусловие	Пользователь зарегистрирован
Шаги	1. Пользователь входит в личный кабинет. 2. Переходит в раздел "Настройки" или "Профиль". 3. Изменяет данные (например, имя). 4. Сохраняет изменения.
Ожидаемый результат	Данные успешно изменены и отображаются корректно.

Требование №2. Поиск мероприятий и фильтрация

Тест-кейс №4: Поиск мероприятий по дате, месту, цене, типу

Описание	Поиск мероприятий по дате, месту, цене, типу
Предусловие	Мероприятия добавлены в систему
Шаги	1. Пользователь переходит на страницу поиска мероприятий. 2. Выбирает фильтры. 3. Нажимает "Поиск".
Ожидаемый результат	Показаны результаты, соответствующие выбранным критериям.

Тест-кейс №5: Сортировка мероприятий по популярности и новизне

Описание	Сортировка мероприятий по популярности и новизне
Предусловие	Список мероприятий доступен
Шаги	1. Пользователь открывает список мероприятий. 2. Выбирает сортировку. 3. Нажимает "Применить".
Ожидаемый результат	Мероприятия отсортированы по выбранному критерию.

Требование №3. Оформление покупки

Тест-кейс №6: Оформление покупки билета через несколько методов оплаты

Описание	Покупка билета через несколько методов оплаты
Предусловие	Мероприятие и билеты доступны
Шаги	1. Пользователь выбирает мероприятие и билет. 2. Переходит к оплате. 3. Выбирает метод оплаты. 4. Подтверждает оплату.
Ожидаемый результат	Билет успешно оплачен, подтверждение отправлено.

Тест-кейс №7: Получение электронного билета после успешной оплаты

Описание	Получение электронного билета после успешной оплаты
Предусловие	Оплата билета завершена
Шаги	1. Пользователь завершает оплату.

	2. Проверяет email или личный кабинет.
Ожидаемый результат	Электронный билет отправлен пользователю.

Требование №4. Система лояльности для постоянных клиентов

Тест-кейс №8: Управление скидками и бонусами для постоянных клиентов

Описание	Управление скидками и бонусами для постоянных клиентов
Предусловие	Пользователь имеет статус постоянного клиента
Шаги	1. Пользователь входит в личный кабинет. 2. Переходит в раздел "Скидки и бонусы". 3. Применяет скидку при покупке билета.
Ожидаемый результат	Скидка или бонус успешно применены при покупке.

Требование №5. Поддержка пользователей

Тест-кейс №9: Возможность чата с поддержкой, отправка заявки на возврат

Описание	Чат с поддержкой и отправка заявки на возврат
Предусловие	Пользователь зарегистрирован
Шаги	1. Пользователь входит в личный кабинет. 2. Переходит в раздел поддержки. 3. Начинает чат или отправляет заявку.
Ожидаемый результат	Заявка успешно отправлена, статус заявки отслеживается.

Требование №6. Уведомления

Тест-кейс №10: Отправка push-уведомлений о новых мероприятиях и скидках

Описание	Push-уведомления о новых мероприятиях и скидках
Предусловие	Новое мероприятие или акция добавлены
Шаги	1. Пользователь подписан на уведомления. 2. Новое мероприятие добавлено. 3. Уведомление отправлено.
Ожидаемый	Пользователь получает уведомление.

результат	

Требование №7. Управление мероприятиями (Администратор)

Тест-кейс №11: Создание нового мероприятия, редактирование информации, удаление мероприятий (Администратор)

Описание	Создание нового мероприятия, редактирование информации и удаление мероприятий администратором
Предусловие	Администратор имеет доступ к панели управления
Шаги	1. Администратор входит в панель управления. 2. Переходит в раздел "Мероприятия". 3. Создает новое мероприятие или редактирует существующее. 4. Удаляет мероприятие, если требуется.
Ожидаемый результат	Мероприятие успешно создано, информация обновлена, мероприятие удалено.

Тест-кейс №12: Добавление и удаление категорий мероприятий (Администратор)

Описание	Добавление и удаление категорий мероприятий администратором
Предусловие	Администратор имеет доступ к панели управления
Шаги	1. Администратор входит в панель управления. 2. Переходит в раздел "Категории мероприятий". 3. Добавляет или удаляет категорию мероприятия.
Ожидаемый результат	Категория успешно добавлена или удалена.

Требование №8. Управление пользователями (Администратор)

Тест-кейс №13: Управление пользователями: изменение статуса, блокировка (Администратор)

Описание	Изменение статуса пользователя (например, постоянный клиент), блокировка пользователя администратором
Предусловие	Администратор имеет доступ к панели управления
Шаги	1. Администратор входит в панель управления. 2. Переходит в раздел "Пользователи". 3. Изменяет статус пользователя или блокирует пользователя.

Ожидаемый результат	Статус пользователя успешно изменен или пользователь заблокирован.

Требование №9. Обработка запросов (Администратор)

Тест-кейс №14: Обработка заявок на возврат билетов и модерация отзывов (Администратор)

Описание	Обработка заявок на возврат билетов и модерация отзывов пользователей администратором
Предусловие	Администратор имеет доступ к панели управления
Шаги	1. Администратор входит в панель управления. 2. Переходит в раздел "Заявки на возврат". 3. Проверяет заявку и подтверждает возврат или отклоняет его. 4. Модерирует отзывы пользователей.
Ожидаемый результат	Заявка на возврат обработана, отзывы модерированы.

Требование №10. Отчеты и аналитика (Администратор)

Тест-кейс №15: Формирование отчетов о продажах, активности пользователей (Администратор)

Описание	Формирование отчетов о продажах и активности пользователей администратором
Предусловие	Администратор имеет доступ к панели управления
Шаги	1. Администратор входит в панель управления. 2. Переходит в раздел "Отчеты". 3. Выбирает нужный отчет и формирует его.
Ожидаемый результат	Отчет успешно сформирован и доступен для просмотра.

Требование №11. Управление безопасностью

Тест-кейс №16: Управление безопасностью

Описание	Управление безопасностью данных и проверка соответствия стандартам (PCI DSS)

Предусловие	Система настроена для управления безопасностью и поддерживает мониторинг стандартов безопасности
Шаги	1. Открыть раздел безопасности системы. 2. Проверить соответствие текущих настроек безопасности стандарту (PCI DSS). 3. Внести изменения в конфигурацию безопасности при необходимости. 4. Применить обновленную политику безопасности. 5. Перезапустить процесс проверки соответствия стандартам.
Ожидаемый результат	Политика безопасности успешно обновлена, проверка соответствия стандартам завершена без ошибок, и система соответствует требованиям безопасности.

Выводы

1. **Функциональные требования:**

   1.1. **Полнота**: Функциональные требования охватывают все основные аспекты системы «Билеты онлайн», включая процессы регистрации пользователей, поиска и бронирования билетов, обработки платежей, интеграции с платёжными системами, управления доступом, а также обеспечения безопасности данных. Каждый из ключевых функциональных процессов имеет соответствующие требования и тест-кейсы, что подтверждает полноту покрытия и проверку заявленных функций.
   1.2. **Достаточность**: Функциональные требования содержат достаточно подробное описание процессов, необходимых для обеспечения нормальной работы системы. Для каждого процесса определены шаги взаимодействия, сценарии возможных ошибок и альтернативных действий. Описанные требования обеспечивают покрытие всех критически важных аспектов системы, таких как бронирование билетов, обработка платежей и обращение в службу поддержки, что позволяет проверить их корректность.

2. **Нефункциональные требования:**

   2.1. **Полнота**: Нефункциональные требования охватывают основные характеристики системы, включая производительность, безопасность, масштабируемость и устойчивость к нагрузкам. Эти аспекты подробно описаны для всех ключевых процессов системы, таких как время

отклика, обработка большого числа запросов и защита данных, что указывает на их достаточную проработанность.

2.2. **Достаточность:** Нефункциональные требования содержат четкие критерии, необходимые для достижения заданных показателей качества системы. Включены подробные спецификации, касающиеся времени отклика для разных операций, допустимых показателей нагрузки и уровней безопасности, что способствует соблюдению стандартов качества и надежности системы.

3. Бизнес-требования:

3.1. **Полнота:** Бизнес-требования направлены на улучшение взаимодействия с пользователями, повышение удобства покупки билетов, поддержку лояльных клиентов и предоставление дополнительных услуг. Эти требования полностью соответствуют целям системы «Билеты онлайн» и поддерживаются функциональными и нефункциональными требованиями, обеспечивая реализацию ключевых бизнес-целей.

3.2. **Достаточность:** Бизнес-требования четко сформулированы и сосредоточены на достижении целей проекта, таких как увеличение продаж билетов, повышение удобства поиска мероприятий и улучшение пользовательского опыта. Они связаны с функциональными требованиями и проверяются через тест-кейсы, что подтверждает их реализацию и соответствие целям проекта.

4. Тест-кейсы:

4.1. **Полнота**: Тест-кейсы охватывают все важные функциональные и нефункциональные аспекты системы, включая сценарии поиска и бронирования билетов, обработки платежей, управления пользовательскими данными, а также обеспечение безопасности. Это подтверждает достаточное тестовое покрытие всех основных функций системы.

4.2. **Достаточность**: Тест-кейсы детализированы и включают проверку основных шагов взаимодействия пользователей с системой. Для каждого тест-кейса определены четкие условия выполнения и критерии приемки, что обеспечивает комплексную проверку корректности функционирования системы и соответствие требованиям.

9. План реализации, тестирования, эксплуатации. Бизнес-план.

Были созданы планы для реализации, тестирования и эксплуатации:

План реализации

1. Введение

- Проект направлен на создание онлайн-системы для бронирования и продажи билетов на концерты, театральные постановки, киносеансы и выставки.
- Целевая аудитория: все пользователи, желающие бронировать и приобретать билеты онлайн.
- Цель проекта: создание платформы с возможностью выбора мест, оплаты, получения билетов по электронной почте и гибкой системы управления клиентскими предпочтениями.

2. Состав команды

- Проектный менеджер: отвечает за контроль всех этапов реализации.
- Разработчики: программирование и настройка системы.
- UI/UX-дизайнер: разработка пользовательского интерфейса.
- Тестировщики: проверка качества и корректности работы системы.
- Специалисты по безопасности: настройка защиты данных.

3. Этапы реализации

1. **Разработка базы данных для хранения информации о пользователях и билетах**:
    - Описание: создание базы данных для управления клиентской информацией, событиями и билетами.
    - Сроки: 1 месяц.
    - Ответственные: разработчики.
2. **Разработка интерфейса для выбора мероприятий и бронирования билетов**:
    - Описание: создание интерфейса, который позволяет пользователям выбирать места, указывать количество билетов, категорию и дополнительные услуги.
    - Сроки: 2 месяца.
    - Ответственные: UI/UX-дизайнер, разработчики.
3. **Интеграция с платёжными системами**:

    - Описание: подключение платёжных методов, таких как электронные кошельки, банковские карты, СБП.
    - Сроки: 1 месяц.
    - Ответственные: разработчики, специалисты по безопасности.
4. **Разработка системы клиентских привилегий**:
    - Описание: разработка системы, позволяющей лояльным клиентам приобретать больше билетов за одно бронирование.
    - Сроки: 1 месяц.
    - Ответственные: разработчики.
5. **Объединение всех компонентов системы и интеграция**:
    - Описание: интеграция всех разработанных модулей в единую систему, проведение внутреннего тестирования.
    - Сроки: 1 месяц.
    - Ответственные: разработчики, тестировщики.

4. Тестирование и отладка

- Описание: тестирование всех функций системы, включая выбор билетов, оплату и получение подтверждений.
- Сроки: 1 месяц.
- Ответственные: тестировщики.

5. Запуск системы

- Описание: запуск системы для конечных пользователей, мониторинг работы системы в режиме реального времени.
- Сроки: 1 неделя.
- Ответственные: вся команда.

План тестирования для продажи онлайн-билетов.

1. Введение

- **Цель тестирования**: Целью данного плана тестирования является проверка всех функциональных и нефункциональных возможностей системы онлайн-продажи билетов для обеспечения соответствия требованиям и стабильной работы системы.
- **Область тестирования**:
    - Регистрация и авторизация пользователей.
    - Выбор мероприятий и бронирование билетов.

- Оформление покупки и оплата.
- Работа системы лояльности для постоянных клиентов.
- Поддержка пользователей.

2. Типы тестирования

1. **Функциональное тестирование**:
   - Проверка регистрации новых пользователей.
   - Тестирование процесса выбора билетов и их бронирования.
   - Проверка корректности обработки оплаты и получения электронных билетов.
2. **Интеграционное тестирование**:
   - Тестирование взаимодействия между разными компонентами системы (модули оплаты, выбора мест, системы лояльности).
3. **Тестирование производительности**:
   - Проверка производительности при высокой нагрузке (большое количество одновременно покупающих пользователей).
   - Оценка времени отклика системы.
4. **Тестирование безопасности**:
   - Проверка защиты личных данных пользователей.
   - Тестирование защиты платежных данных и выполнения транзакций.

3. Сценарии тестирования

1. **Сценарий: Регистрация нового пользователя**:
   - Проверка процесса регистрации нового пользователя, включая отправку данных для входа по электронной почте.
2. **Сценарий: Бронирование билетов**:
   - Тестирование выбора мероприятий, бронирования билетов и подтверждения бронирования.
3. **Сценарий: Оплата**:
   - Проверка корректности оформления заказа и выполнения платежа.
4. **Сценарий: Система лояльности**:
   - Тестирование работы системы скидок и привилегий для постоянных клиентов.
5. **Сценарий: Поддержка пользователей**:
   - Проверка работы службы поддержки и обработки пользовательских запросов.

4. Критерии приемки

- Все функциональные возможности системы работают корректно без задержек.
- Система способна поддерживать работу с минимальной задержкой для 1000 пользователей одновременно.
- Все платежи обрабатываются корректно и безопасно.

5. Риски

- Проблемы с безопасностью: Необходимо тестировать механизмы безопасности, чтобы избежать утечек личных данных.
- Проблемы с производительностью: Увеличение нагрузки может привести к замедлению работы системы.
- Проблемы с интеграцией: Взаимодействие между компонентами системы может вызвать ошибки.

6. План выполнения тестирования

- **Подготовка тестового окружения**: Создание тестовой базы данных и инструментов для выполнения тестов.
- **Функциональное тестирование**: Проверка корректности выполнения ключевых функций системы.
- **Интеграционное тестирование**: Тестирование взаимодействия между разными модулями системы.
- **Тестирование производительности**: Проверка работы системы при пиковых нагрузках.
- **Тестирование безопасности**: Проверка защиты данных и платежных транзакций.
- **Пользовательское тестирование**: Привлечение реальных пользователей для тестирования удобства интерфейса.

7. Отчетность

- **Еженедельные отчеты**: Формирование отчетов о результатах тестирования, обнаруженных ошибках и их исправлении.
- **Финальный отчет**: Подготовка заключительного отчета о тестировании и рекомендациях по улучшению системы.

8. Заключение

При соблюдении данного плана тестирования система получится устойчивой к нагрузкам, интуитивно понятной.

10. План эксплуатации для системы продажи онлайн-билетов

1. Введение

- **Цель и область применения системы**: Система предназначена для автоматизации процесса бронирования и продажи билетов на различные мероприятия, такие как концерты, театральные постановки, киносеансы и выставки.
- **Описание системы**: Система включает в себя функционал для выбора мероприятий, бронирования мест, оплаты билетов и получения их по электронной почте.

2. Административное управление

- **Назначение ролей**:
    - Администратор системы: отвечает за управление пользователями и событиями, настройку параметров системы и контроль за её работой.
    - Пользователи (клиенты): могут бронировать и покупать билеты на доступные мероприятия.
- **Управление событиями и билетами**:
    - Администратор добавляет новые мероприятия и указывает детали: место, дату, цену билетов.
    - Обновление информации о доступных местах и ценах.

3. Управление пользователями

- **Регистрация и авторизация пользователей**: Поддержка пользователей при создании учетных записей и авторизации.
- **Система привилегий и лояльности**: Управление уровнем лояльности пользователей и настройка привилегий для постоянных клиентов (возможность покупки большего количества билетов, скидки и т.д.).

4. Работа с финансовыми транзакциями

- **Интеграция с платёжными системами**: Поддержка оплаты через банковские карты, электронные кошельки и СБП.
- **Управление возвратами**: Поддержка возвратов средств при отмене бронирований в рамках установленных правил.

5. Поддержка клиентов

- **Служба поддержки**: Организация службы поддержки для обработки запросов пользователей по вопросам бронирования, оплаты и возвратов.
- **Часто задаваемые вопросы (FAQ)**: Создание раздела с наиболее частыми вопросами для снижения нагрузки на службу поддержки.

6. Обеспечение безопасности и доступности

- **Аутентификация и авторизация**: Реализация механизмов для безопасной авторизации пользователей.
- **Шифрование данных**: Защита персональных и платежных данных пользователей через шифрование.
- **Регулярное обновление системы**: Проведение обновлений системы для устранения уязвимостей и добавления новых функций.

7. Обучение и поддержка персонала

- **Обучающие материалы**: Разработка инструкций и обучающих материалов для администраторов системы.
- **Техническая поддержка**: Предоставление команды поддержки для решения технических вопросов в процессе эксплуатации системы.

8. Обновления и поддержка системы

- **План обновлений**: Регулярное обновление функциональности системы, внедрение новых модулей и функций по мере необходимости.
- **Мониторинг системы**: Постоянный мониторинг работы системы для обнаружения и устранения проблем.

9. Резервное копирование и восстановление данных

- **Резервное копирование**: Ежедневное резервное копирование данных о пользователях, бронированиях и транзакциях.
- **План восстановления**: Разработка процедур восстановления системы в случае сбоя или потери данных.

10. Аудит и анализ работы системы

- **Регулярные аудиты безопасности**: Проведение проверок на соответствие стандартам безопасности.
- **Оценка производительности**: Мониторинг производительности системы под нагрузкой и её оптимизация.

11. Заключение

- Подведение итогов и обзор ключевых аспектов плана эксплуатации.
- Указание на дальнейшие шаги и улучшения в развитии системы.

Бизнес-план

1. Объем инвестиций

- **Разработка программного обеспечения**: 5 000 000 рублей
- **Инфраструктура (серверы, хостинг, оборудование)**: 3 500 000 рублей
- **Маркетинг и продвижение**: 1 000 000 рублей
- **Операционные расходы (зарплаты команды и тд)**: 2 500 000 рублей
- **Непредвиденные расходы**: 500 000 рублей
- **Итого**: 12 500 000 рублей

2. Описание продукта

- Система онлайн-бронирования и продажи билетов предоставляет пользователям возможность выбирать и бронировать места на концерты, театральные постановки, киносеансы и выставки.
- В системе реализована программа лояльности для постоянных клиентов, что позволяет им бронировать больше билетов с дополнительными привилегиями и скидками.

3. Целевая аудитория

- Широкий круг пользователей, посещающих культурные мероприятия, включая концерты, кино, театры и выставки.
- Организаторы мероприятий, желающие автоматизировать процесс продажи билетов и улучшить взаимодействие с клиентами.

4. Конкурентные преимущества

- **Интуитивный интерфейс**: Легкость использования для всех категорий пользователей.
- **Система лояльности**: Возможность предоставления скидок и привилегий для постоянных клиентов.
- **Разнообразие платёжных методов**: Поддержка банковских карт, электронных кошельков и СБП.
- **Интеграция с внешними системами**: Возможность подключения сторонних сервисов для аналитики и маркетинга.
- **Удобство для организаторов**: Простой интерфейс для управления мероприятиями, категориями билетов и отчетами по продажам.

5. Период окупаемости

- **Ожидаемый ежемесячный доход**:
    - Продажа билетов: 2 000 000
    - Комиссионные от организаторов: 500 000
    - Итого: 2 500 000
- **Чистая прибыль в месяц**: 1 500 000
- **Период окупаемости**: 12 500 000 / 1 500 000 = 8,5 месяцев

6. Рентабельность

- **Годовая чистая прибыль**: 1 500 000 * 12 = 18 000 000
- **ROI (Коэффициент окупаемости инвестиций)**:
    - (Чистая прибыль - Объем инвестиций) / Объем инвестиций
    - (18 000 000 - 12 500 000) / 12 500 000 = 44%

План сопровождения

1. **Гибкое тестирование на этапе приёмки** – настройка программного обеспечения под конкретную инфраструктуру клиента, включая проверку совместимости с физическими компонентами и условиями эксплуатации.

2. **Организация системы мониторинга ошибок** – разработка механизма для отслеживания сбоев и анализа ошибок, а также установление эффективных каналов взаимодействия с клиентами для своевременного информирования и решения проблем.
3. **Передача клиенту всей необходимой документации** – предоставление полного комплекта материалов по эксплуатации и управлению системой, включая руководство пользователя и инструкции по администрированию.

4. **Применение инструментов управления версиями** – использование автоматизированных средств для контроля версий и интеграции обновлений, чтобы обеспечить упрощенный процесс внедрения новых версий ПО.

5. **Документирование всех изменений в системе** – составление отчетов о внесенных исправлениях, новых функциях и результатах тестирования, а также предоставление обновленной документации клиенту после каждого релиза.

11. План ревизии проекта "Билеты онлайн"

Цель ревизии — проанализировать изменения в проекте "Билеты онлайн" с момента его первоначального планирования до текущего состояния. Рассмотрим основные обновления, включая изменения в подходе к разработке, архитектуре и функциональности.

1. Изменения в модели жизненного цикла

- **Изначально**: Проект разрабатывался на основе **спиральной модели** с элементами Agile, которая предполагала поэтапную проработку требований и оценку рисков.

- **Сейчас**: Переход на **инкрементную модель** позволил поэтапно внедрять готовые части системы, ускоряя процесс тестирования и улучшая адаптацию к изменениям в требованиях заказчика.

2. Изменения в функциональных возможностях

- **Личный кабинет**:
  - **Изначально**: Личный кабинет позволял пользователям просматривать историю покупок и редактировать личные данные.
  - **Сейчас**: Добавлена возможность управления настройками уведомлений и выгрузка билетов в формате PDF, что улучшило пользовательский опыт.
- **Система рекомендаций**:
  - **Изначально**: Система рекомендаций была ограничена базовыми фильтрами по дате и типу мероприятий.
  - **Сейчас**: Внедрены алгоритмы, учитывающие предпочтения пользователей, основанные на их истории покупок и активности, что позволило более точно предлагать интересующие мероприятия.

3. Изменения в нефункциональных требованиях

- **Производительность**:
  - **Изначально**: Поддержка работы до 10 000 пользователей в пиковые моменты.
  - **Сейчас**: Улучшена масштабируемость за счёт оптимизации кэширования и использования балансировщиков нагрузки, что увеличило поддержку до 15 000 пользователей.
- **Безопасность**:
  - **Изначально**: Реализована базовая защита данных в соответствии с PCI DSS.
  - **Сейчас**: Внедрены дополнительные меры, такие как двухфакторная аутентификация и мониторинг подозрительных активностей, что повысило уровень безопасности.

4. Изменения в поддержке пользователей

- **Изначально**: Поддержка пользователей была реализована через систему FAQ и форму обратной связи.
- **Сейчас**: Добавлен **онлайн-чат**, позволяющий пользователям получать помощь в реальном времени. Это повысило оперативность решения проблем и увеличило удовлетворённость клиентов.

5. Выводы

- Переход на инкрементную модель разработки улучшил процесс внедрения новых функций и ускорил тестирование.
- Оптимизация масштабируемости помогла улучшить работу системы под нагрузкой, обеспечив стабильную работу с увеличением числа пользователей.
- Обновления в функционале и усиление безопасности сделали продукт более привлекательным и надежным для пользователей.

Эти изменения позволили проекту "Билеты онлайн" стать более гибким и готовым к изменяющимся требованиям, улучшив качество обслуживания и удовлетворенность пользователей.

12. Итоговые результаты работы

В ходе проделанной работы были разработаны нормативные документы для реализации системы продажи и бронирования онлайн-билетов “Билеты-онлайн”. Составлены основные документы, такие как: техническое задание с основным списком требований, выбрана модель жизненного цикла и методологии разработки, диаграммы прецедентов, состояний и классов. Помимо этого была выбрана архитектура нашей системы, составлена матрица трассировки требований и планы реализации, тестирования разрабатываемого проекта. Описаны четко и детально все требования заказчика к исполнителю работ.

13. Дальнейшие планы развития

В качестве основных планов для развития системы можно выделить несколько направлений:

Расширение функциональности:

- Внедрение обновленной системы обратной связи для пользователей с возможностью оценки мероприятий и услуг.
- Интеграция с другими системами, такими как социальные сети или сторонние платежные сервисы.

Улучшение интерфейса:

- Оптимизация UX/UI для повышения удобства и интуитивности использования системы.
- Добавление персонализированных рекомендаций на основе интересов и предпочтений пользователей.

Расширение отчетности и аналитики:

- Доработка системы отчетов, включающей аналитику по продажам и посещаемости мероприятий.
- Интеграция с инструментами бизнес-аналитики для визуализации данных.

Повышение безопасности:

- Внедрение многофакторной аутентификации для усиления защиты пользовательских данных.
- Улучшение мер по защите персональной информации и предотвращению мошенничества.

Планирование и управление ресурсами:

- Разработка модуля для управления ресурсами, такими как залы, билеты и расписание мероприятий.
- Внедрение инструмента для планирования маркетинговых акций и учета затрат на продвижение.

Техническая поддержка и обучение:

- Создание обучающих материалов и ресурсов для пользователей системы.
- Организация службы поддержки для оперативного решения вопросов и проблем пользователей.

Инновации и новые технологии:

- Использование искусственного интеллекта для анализа данных и персонализированных предложений.
- Разработка рекомендательных алгоритмов для повышения продаж и улучшения пользовательского опыта.