1)Название продукта

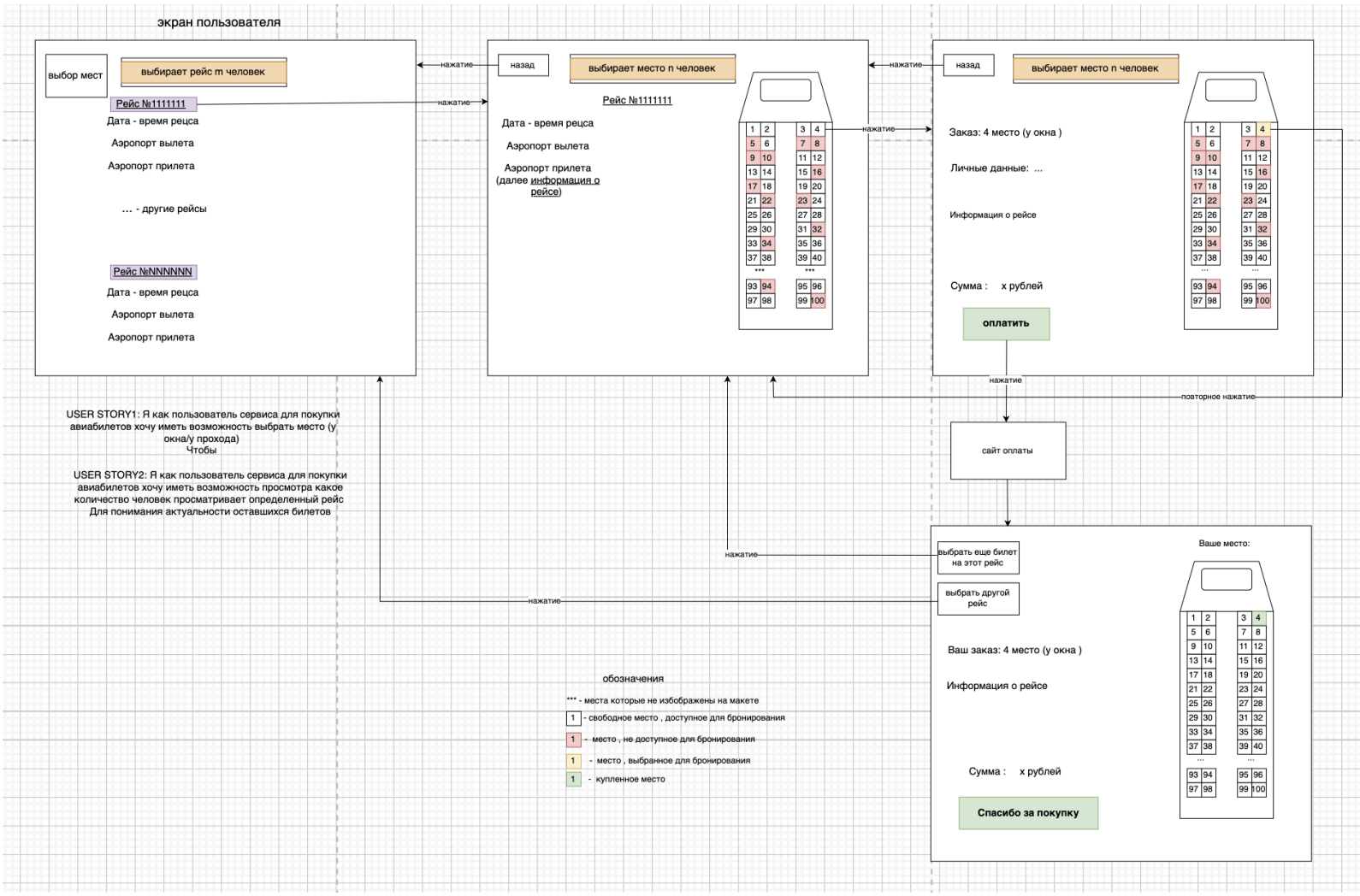
Онлайн сервис для покупки авиабилетов

2)User Story

USER STORY 1: Я как пользователь сервиса для покупки авиабилетов хочу иметь возможность выбрать место (у окна/у прохода), для того чтобы мне было комфортнее провести полет

USER STORY 2: Я как пользователь сервиса для покупки авиабилетов хочу иметь возможность просмотра количества человек просматривающих определенный рейс в данный момент времени
Для понимания актуальности оставшихся билетов

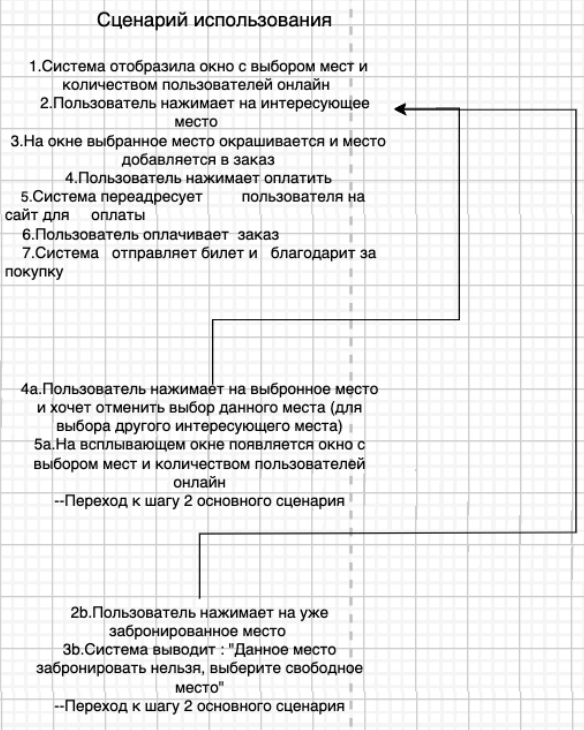
3)Макет



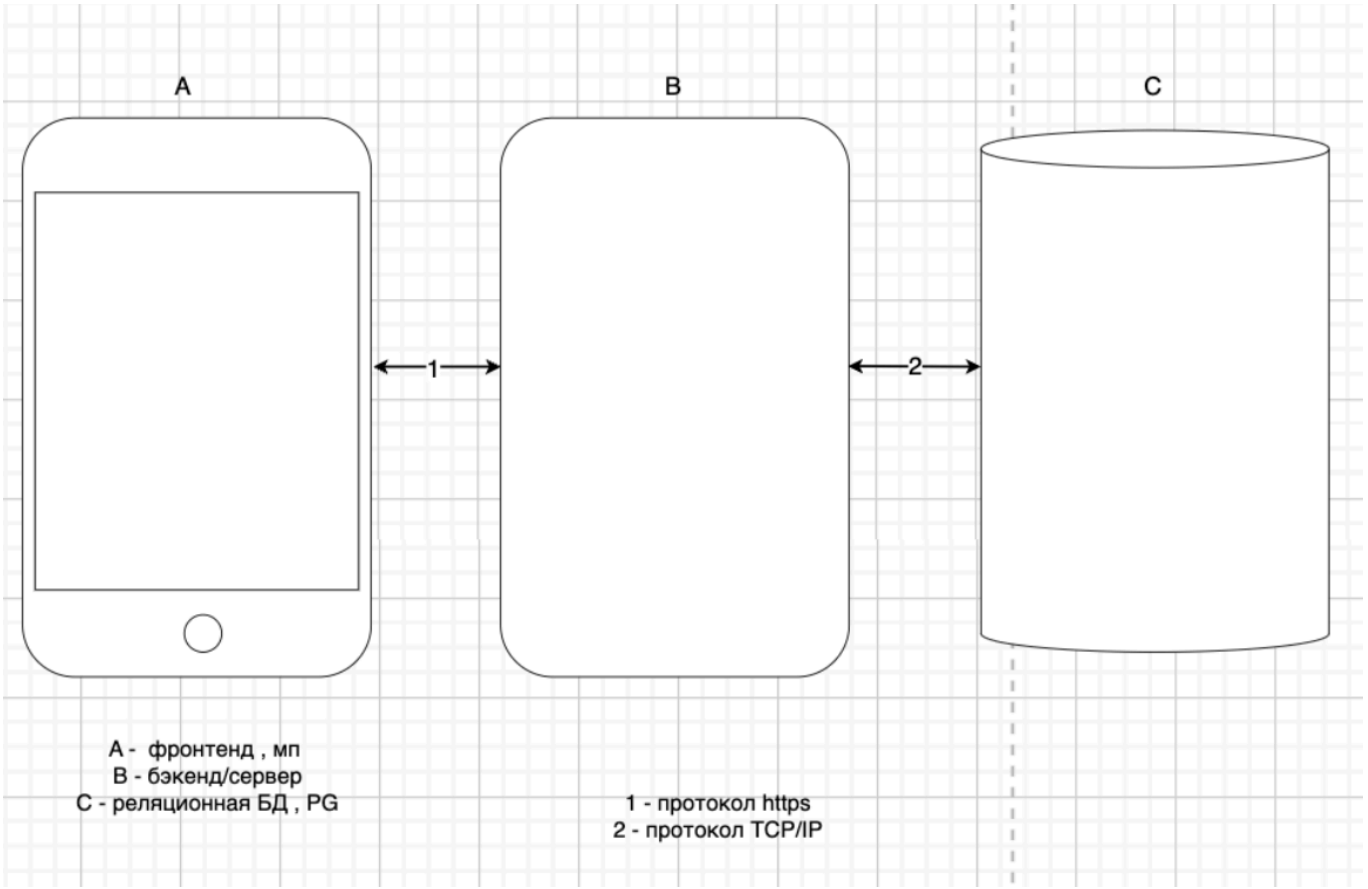
4)Use Case

Use Case

Название поля	Что содержит	Действие
Заголовок	Краткое и ясное название сценария, описание задачи	Пользователь оплачивает выбранное место
Акторы	Кто взаимодействует с системой. Пользователи и внешние системы	Пользователь приложения
Предусловие	В каком состоянии должен находиться каждый участник, чтобы юзкейс произошел	Пользователь зашел на сайт авиакомпании
Ограничения	Любые огранич. которые могут повлиять на выполнение сценария	Нельзя выбрать занятые места
Триггер	Что провоцирует начало сценария	Пользователь зашел в раздел "Выбор места"
Основной сценарий	Основная последовательность действий, когда все идет по плану и пользователь достигает успеха	1.Система отобразила окно с выбором мест и количеством пользователей онлайн 2.Пользователь нажимает на интересующее место 3.На окне выбранное место окрашивается и место добавляется в заказ 4.Пользователь нажимает оплатить 5.Система переадресует пользователя на сайт для оплаты 6.Пользователь оплачивает заказ 7.Система отправляет билет и благодарит за покупку
Альтернативный сценарий	Что происходит, если процесс отклоняется от основного сценария, но также достиг успеха	4a.Пользователь нажимает на выбронное место и хочет отменить выбор данного места (для выбора другого интересующего места) 5a.На всплывающем окне появляется окно с выбором мест и количеством пользователей онлайн --Переход к шагу 2 основного сценария
Исключительный сценарий	Что происходит , когда возникла ошибка и цель не может быть достигнута	2b.Пользователь нажимает на уже забронированное место 3b.Система выводит : "Данное место забронировать нельзя, выберите свободное место" --Переход к шагу 2 основного сценария
критерий успеха	когда "действие" считается успешным	Основной сценарий: Пользователь купил билет с необходимым местом Альтернативный сценарий: Пользователь отменил выбор места



4)Архитектура



4.1 Frontend

Технологии: React, Redux, Axios для управления состоянием и выполнения HTTP-запросов.

Функции: Отображение доступных рейсов и мест.

Обработка ввода пользователя.

Отображение статуса платежа.

4.2 Backend

Технологии: Node.js, Express

Функции: Обработка запросов от клиента.

Управление данными о рейсах и местах.

Взаимодействие с системой платежей.

4.3 БД

Технологии: PostgreSQL для работы с базой данных.

Функции: хранение данных.

5)Логика выбора билетов

5.1 Выбор места

Система будет отображать план самолета с местами, где пользователи могут выбрать свободные места.

Места делятся на стандартные и места у окна. Места у окна будут отображаться с более высокой ценой.

Реализация выбора мест будет основана на использовании RESTful API, который будет обрабатывать запросы на выбор мест и подтверждение бронирования.

6) Система платежей

Система платежей уже подключена. При успешной оплате система получает уведомление о статусе платежа и, в случае успешной оплаты, резервирует выбранные места.

7) Нагрузка на сервис

Количество самолетов: 10

Количество рейсов в день: 20

Количество мест в одном рейсе: 100

Максимальное количество пользователей: 500

При максимальной загрузке (20000 пользователей) система должна обрабатывать до 20000 запросов на продажу билетов одновременно. Использование технологий, таких как кэширование (Redis) и Load Balancer, поможет распределить нагрузку и обеспечить быструю обработку запросов.

Система должна быть доступна 99% времени

Требования производительности:

Окно с выбором мест должно открываться не более 2 секунд.

Запрос на получение информации о месте по уникальному идентификатору должен выдерживать нагрузку 1 rps.

8) Обоснование выбора стека

Node.js и Express: Позволяют быстро разрабатывать серверные приложения, обрабатывать множество соединений одновременно.

PostgreSQL: Подходит для хранения динамических данных, таких как информация о рейсах и пользователях.

React: Позволяет создавать интерактивные пользовательские интерфейсы и обеспечивает хорошую производительность.

Redux: Управляет состоянием приложения, что особенно полезно для сложных интерфейсов с динамическими данными.

Данный стек обеспечивает масштабируемость, высокую производительность и легкость в разработке, что критично для системы, обрабатывающей большое количество пользователей и запросов.

9)Проектирование:

Авиакомпания

- ID (PK)
- Название
- Количество самолетов
- Количество рейсов в день

Самолет

- ID (PK)
- Модель
- Вместимость
- ID_Авиакомпании (FK)

Рейс

- ID (PK)
- Номер рейса
- Дата и время вылета
- ID_Самолета (FK)
- Место назначения
- Количество доступных мест

Пассажир

- ID (PK)
- Имя
- Фамилия
- Email
- Телефон

Бронирование

- ID (PK)
- ID_Рейса (FK)
- ID_Пассажира (FK)
- Дата бронирования
- Статус

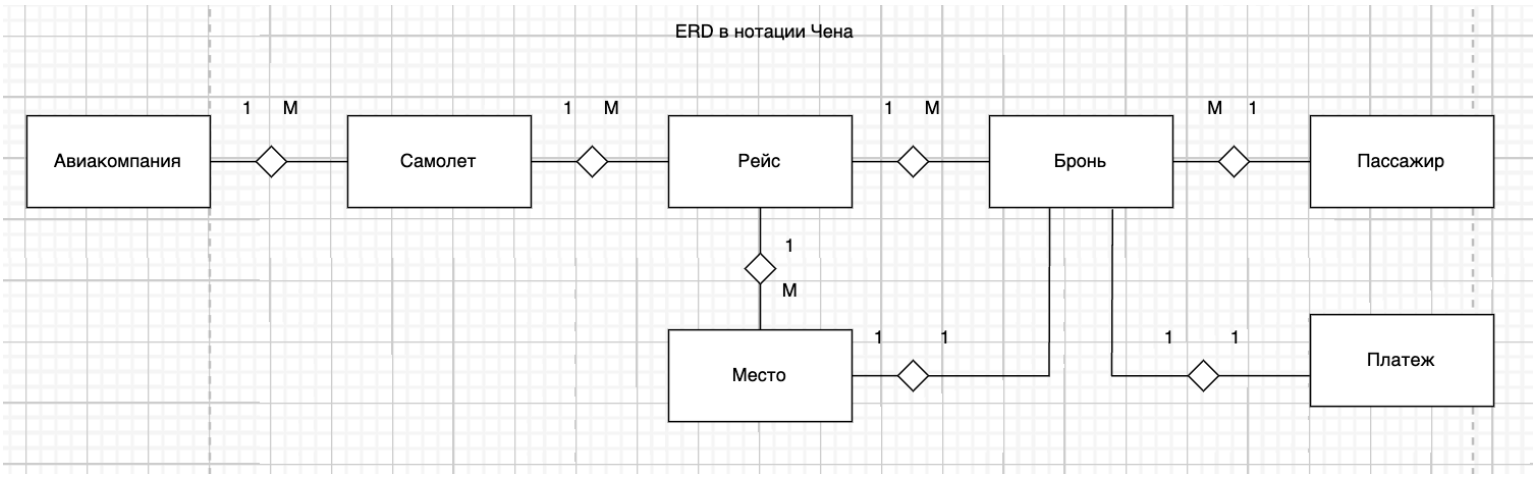
Место

- ID (PK)
- Номер места
- ID_Рейса (FK)
- Статус
- Услуга

Платеж

- ID (PK)
- Сумма
- Дата
- ID_Бронирования (FK)
- Статус

Реализация ERD



10) Функции

Backend:
1)Управление рейсами и самолетами
Модуль для управления данными о рейсах (создание, обновление, удаление рейсов).
2)Управление пользователями
Управление сессиями (отслеживание активных пользователей).
3)Бронирование билетов
Проверка доступных мест на рейсах.
4)Управление платежами
Интеграция с платежной системой для обработки платежей (например, через API).
Обработка успешных и неуспешных платежей.
Уведомление пользователей о статусе их транзакции.
5)API для фронтенда
RESTful API для взаимодействия с фронтендом (например, для получения данных о рейсах, пользователях и билетах).

Frontend:
1)Интерфейс пользователя
Главная страница с информацией о доступных рейсах.
Страница выбора места (включая возможность выбора мест у окна).
2)Отображение доступных мест
Визуализация схемы самолета (например, 2 по 2 кресла) с возможностью выбора мест.
Отображение информации о доступных и забронированных местах.
3)Процесс покупки билета
Форма для ввода информации о пассажирах.
Интеграция с платежной системой для обработки транзакций.
4)Уведомления и сообщения
Уведомления о статусе бронирования (успешно/неуспешно).
Информация о купленном билете
5)Адаптивный дизайн
Обеспечение корректного отображения на различных устройствах (мобильные телефоны, планшеты, настольные ПК).
6)Обработка ошибок
Уведомления об ошибках (например, при неуспешной попытке купить билет).
Логирование ошибок для последующего анализа и устранения.
Уведомление об ошибке при попытке забронировать уже занятое место

Возможна интеграция CRM системы в сервис для управления клиентами