Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение   
высшего образования

**Иркутский национальный исследовательский технический университет**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

Допускаю к защите

|  |  |
| --- | --- |
| Руководитель |  |
|  | подпись |
|  | М.Д. Каташевцев |
|  | И.О. Фамилия |

Разработка фуллстек-cервиса с использованием Django и Vue.js

|  |
| --- |
| наименование темы |

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе по дисциплине

|  |
| --- |
| WEB-программирование |

|  |
| --- |
| 1.006.00.00 - ПЗ |
| обозначение документа |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | ИСТб-22-2 |  |  |  | | В.С. Герасимова |
|  |  | шифр |  | подпись |  | | И.О. Фамилия |
| Нормоконтроль |  |  |  |  |  | | М.Д. Каташевцев |
|  |  |  |  | подпись |  | | И.О. Фамилия |
| Курсовой проект защищен с оценкой | | | |  | |  | |

Иркутск 2024 г.

Содержание

[Введение 3](#_Toc185204809)

[1 Описание предметной области 4](#_Toc185204810)

[2 Функциональные возможности 6](#_Toc185204811)

[3 Проектирование веб-приложения 7](#_Toc185204812)

[4 Иерархия компонентов 19](#_Toc185204813)

[5 Проектирование интерфейса 20](#_Toc185204814)

[6 Функции, доступные на сайте 28](#_Toc185204815)

[7 Отличительная особенность 29](#_Toc185204816)

[Заключение 30](#_Toc185204817)

[Список использованных источников 31](#_Toc185204818)

# **Введение**

Цель данной курсовой работы — закрепить теоретические знания, полученные в рамках курса «WEB-программирование», и развить практические навыки их применения при решении конкретной задачи: создания веб-приложения. Работа направлена на формирование умений принимать обоснованные решения в процессе проектирования и получения базового опыта разработки программного обеспечения.

Для реализации проекта необходимо разработать веб-приложение, используя современные технологии: Django для серверной части (бэкенда) и Vue.js для клиентской части (фронтенда). Эти инструменты выбраны благодаря их популярности, гибкости и способности эффективно взаимодействовать при создании сложных и масштабируемых приложений.

Django — это высокоуровневый фреймворк для разработки веб-приложений на Python, предлагающий широкий набор встроенных инструментов, таких как системы аутентификации, работа с базами данных, маршрутизация запросов и другие функции, которые упрощают процесс разработки.

Vue.js — прогрессивный JavaScript-фреймворк, предназначенный для создания пользовательских интерфейсов. Он выделяется простотой изучения, высокой производительностью и модульным подходом к построению интерфейсов.

# **1 Описание предметной области**

В рамках проекта разработан специализированный сайт, предназначенный для управления данными, связанными с авиационной сферой. Он обеспечивает удобный доступ к информации о рейсах, клиентах, багаже, билетах и самолетах, что позволяет эффективно решать задачи, возникающие в деятельности авиакомпаний и аэропортов.

Предметная область включает:

* Клиентов: пассажиры, зарегистрированные в системе. Хранятся их контактные данные и информация о связях с другими объектами (например, билетами и багажом).
* Рейсы: сведения о текущих, запланированных и завершенных рейсах, включая маршрут, время отправления/прибытия, статус и используемый самолет.
* Билеты: информация о билетах на рейсы, их статусе и привязке к пассажирам и рейсам.
* Багаж: зарегистрированный багаж с указанием веса, размеров и принадлежности конкретному пассажиру.
* Самолеты: данные о воздушных судах, включая модели, технические характеристики и назначение на конкретные рейсы.

Основной целью создания сайта является автоматизация процессов учета и управления данными. Это включает упрощение поиска информации о рейсах, клиентах, багаже и билетах, повышение точности и эффективности внесения и редактирования данных, предоставление инструментов фильтрации и сортировки для быстрой работы с информацией.

На рисунке 1 представлена концептуальная модель предметной области на основе метода «Объект-Связь».

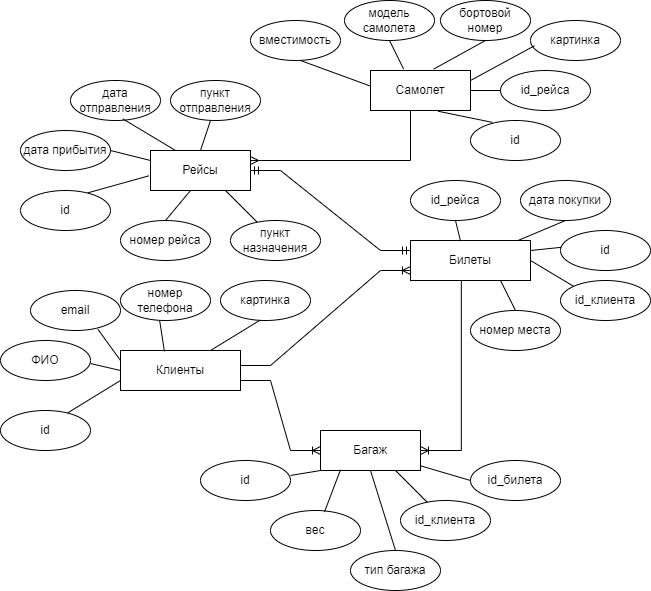


Рисунок 1 – ER-диаграмма предметной области

Исходя из предметной области были выделены следующие сущности:

1. Клиенты;
2. Рейсы;
3. Самолеты;
4. Багаж;
5. Билеты.

В таблице 1 описаны сущности предметной области.

Таблица 1 – Описание сущностей предметной области

|  |  |
| --- | --- |
| **Название** | **Описание** |
| Клиенты | Хранятся данные о пассажирах, включая их имя, контактную информацию (телефон, электронная почта) и связь с другими объектами, такими как билеты и багаж. Клиенты являются ключевыми участниками системы. |
| Рейсы | Содержат информацию о полетах, включая номер рейса, маршрут (пункт отправления и назначения), время вылета и прилета. |
| Самолеты | Хранится информация о воздушных судах, таких как модель самолета, вместимость (количество пассажиров), максимальная грузоподъемность. Самолеты назначаются на конкретные рейсы. |
| Багаж | Регистрация информации о багаже пассажиров: номер багажа, вес, габариты и привязка к конкретному пассажиру и рейсу. Эта сущность позволяет отслеживать багаж в системе. |
| Билеты | Описываются параметры билетов: номер места, дата покупки и привязка к конкретному рейсу и пассажиру. Билеты являются связующим звеном между рейсом и клиентом. |

# **2 Функциональные возможности**

Приложение должно работать с базой данных, в которой хранится информация о сущностях предметной области.

Веб-приложение должно иметь серверную часть (бэкенд) и клиентскую часть (фронтенд).

Необходимо, чтобы приложение выполняло следующие требования:

1. Взаимодействие с базой данных;
2. Осуществление основных CRUD операций для всех моделей предметной области (добавление, удаление, редактирование, чтение);
3. У каждой записи должно быть изображение, которое пользователь может устанавливать для новых записей, заменять у уже существующих записей, при клике по изображению должно открываться модальное окно;
4. Данные должны быть привязаны к пользователю, который из вносит, и каждый пользователь должен иметь возможность просматривать только те данные, которые он внес;
5. Должна иметься возможность фильтрации записей по всем смысловым столбцам;
6. Должна иметься страница авторизации в систему;
7. Возможность экспортировать данные основной таблицы в Excel или Word.

# **3 Проектирование веб-приложения**

3.1 Проектирование базы данных

При переходе от семантической модели данных к физической модели базы данных была создана модель с использованием инструмента DbDesigner, представленная на рисунке 2.

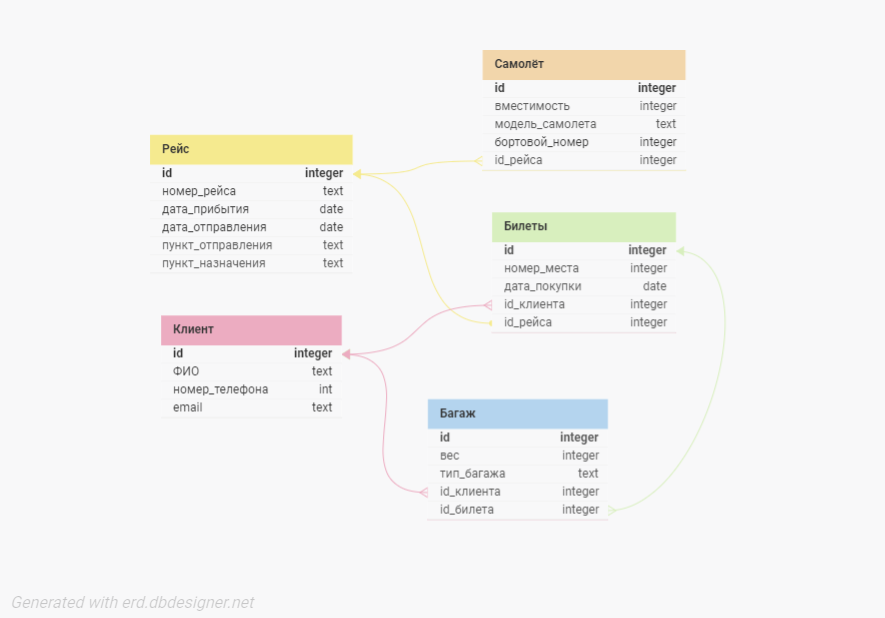


Рисунок 2 - Физическая модель базы данных

3.2 Проектирование программного интерфейса

API — это интерфейс для взаимодействия между программами, представляющий собой набор методов и правил, позволяющих приложениям обмениваться данными и выполнять запросы.

Через API программа может запрашивать информацию из базы данных или инициировать выполнение определенных операций.

В таблице 2 представлено описание API приложения.

Таблица 2 – Описание API

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| URL | Метод | Назначение | JSON вход | JSON выход |
| /api/clients/ | GET | API для получения списка клиентов | - |  |
| POST | Добавление клиента |  |  |
| URL | Метод | Назначение | JSON вход | JSON выход |
| /api/clients/${id}/ | DELETE | Удаление клиента | - | - |
| PUT | Редактирование данных клиента |  |  |
| /api/clients/export?type=${type} | GET | Загрузка данных в Word/Excel | - | - |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| URL | Метод | Назначение | JSON вход | JSON выход |
| /api/flights/ | GET | API для получения списка рейсов | - |  |
| POST | Добавление рейса |  |  |
|  |  |  |  |  |
| URL | Метод | Назначение | JSON вход | JSON выход |
| /api/flights/${id}/ | DELETE | Удаление рейса | - | - |
| PUT | Редактирование данных рейса |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| URL | Метод | Назначение | JSON вход | JSON выход |
| /api/tickets/ | GET | API для получения списка билетов | - |  |
| POST | Добавление билета |  |  |
| URL | Метод | Назначение | JSON вход | JSON выход |
| /api/tickets/${id}/ | DELETE | Удаление билета | - | - |
| PUT | Редактирование данных билета |  |  |
| URL | Метод | Назначение | JSON вход | JSON выход |
| /api/baggage/ | GET | API для получения информации о багаже | - |  |
| POST | Добавление багажа |  |  |
| URL | Метод | Назначение | JSON вход | JSON выход |
| /api/baggage/${id}/ | DELETE | Удаление багажа | - | - |
| PUT | Редактирование данных багажа |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| URL | Метод | Назначение | JSON вход | JSON выход |
| /api/airplanes/ | GET | API для получения информации о самолетах | - |  |
| POST | Добавление самолета |  |  |
| URL | Метод | Назначение | JSON вход | JSON выход |
| /api/airplanes/${id}/ | DELETE | Удаление самолета | - | - |
| PUT | Редактирование данных самолета |  |  |
| /api/user/info/ | GET | API для получения информации о пользователе | - |  |
| /api/user/login/ | POST | Вход пользователя | { "username": "admin", "password": "123" } |  |
| /api/auth/otp-login/ | POST | Вход с использованием OTP | { "key": "<OTP ключ>" } |  |
| URL | Метод | Назначение | JSON вход | JSON выход |
| /api/auth/otp-status/ | GET | Проверка статуса OTP для пользователя | - |  |
| /api/auth/otp-qr-code/ | GET | Генерация QR-кода для OTP | - | Изображение в формате PNG |

# **4 Иерархия компонентов**

На рисунке 3 представлена иерархия компонентов приложения.

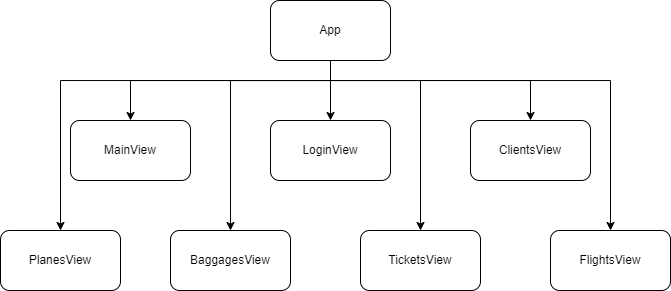


Рисунок 3 – Иерархия компонентов

# **5 Проектирование интерфейса**

При проектировании интерфейса была разработана информационная архитектура сайта (см. рис. 4).



Рисунок 4 – Информационная архитектура интерфейса

Разработанный интерфейс страниц представлен на рисунках 5-19.

Главная страница

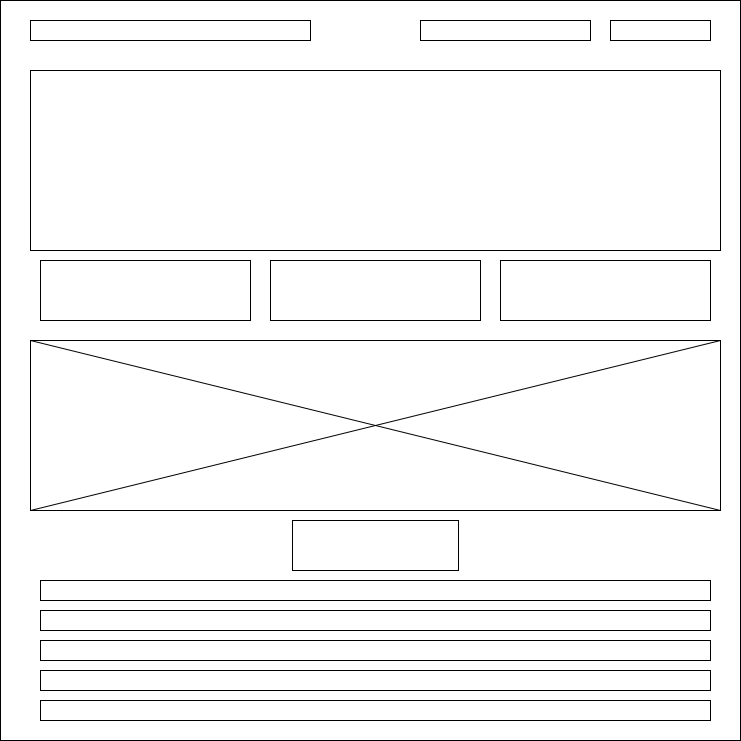


Рисунок 5 – Макет страницы «Главная»

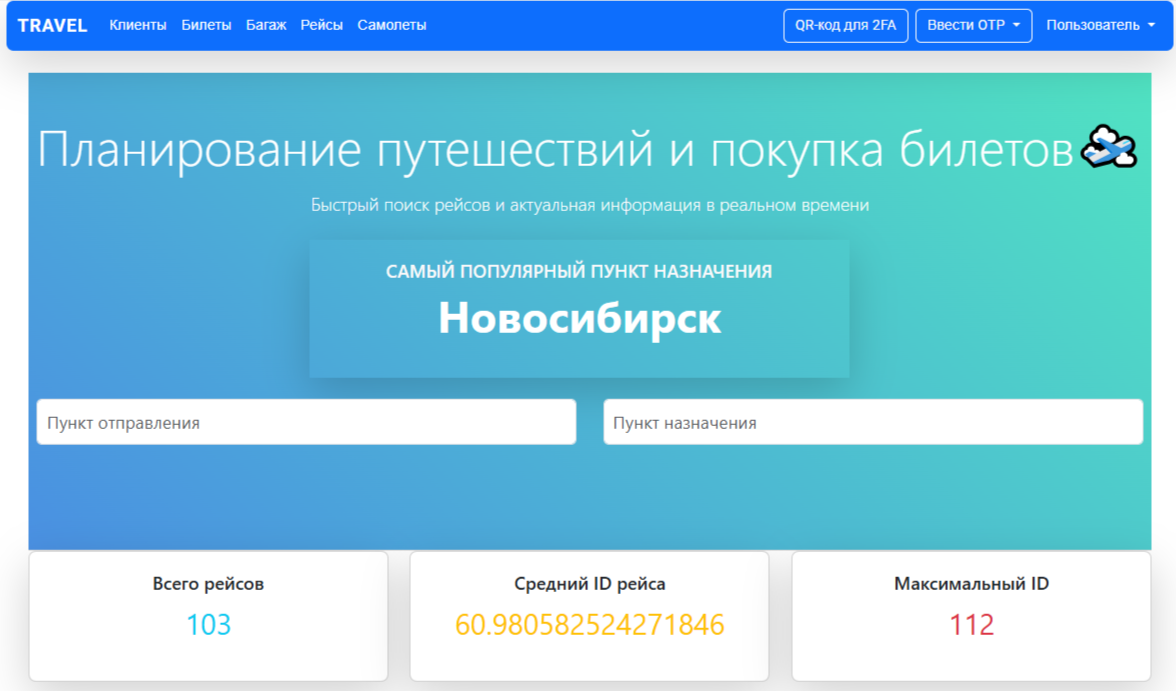


Рисунок 6 – Первый фрагмент страницы

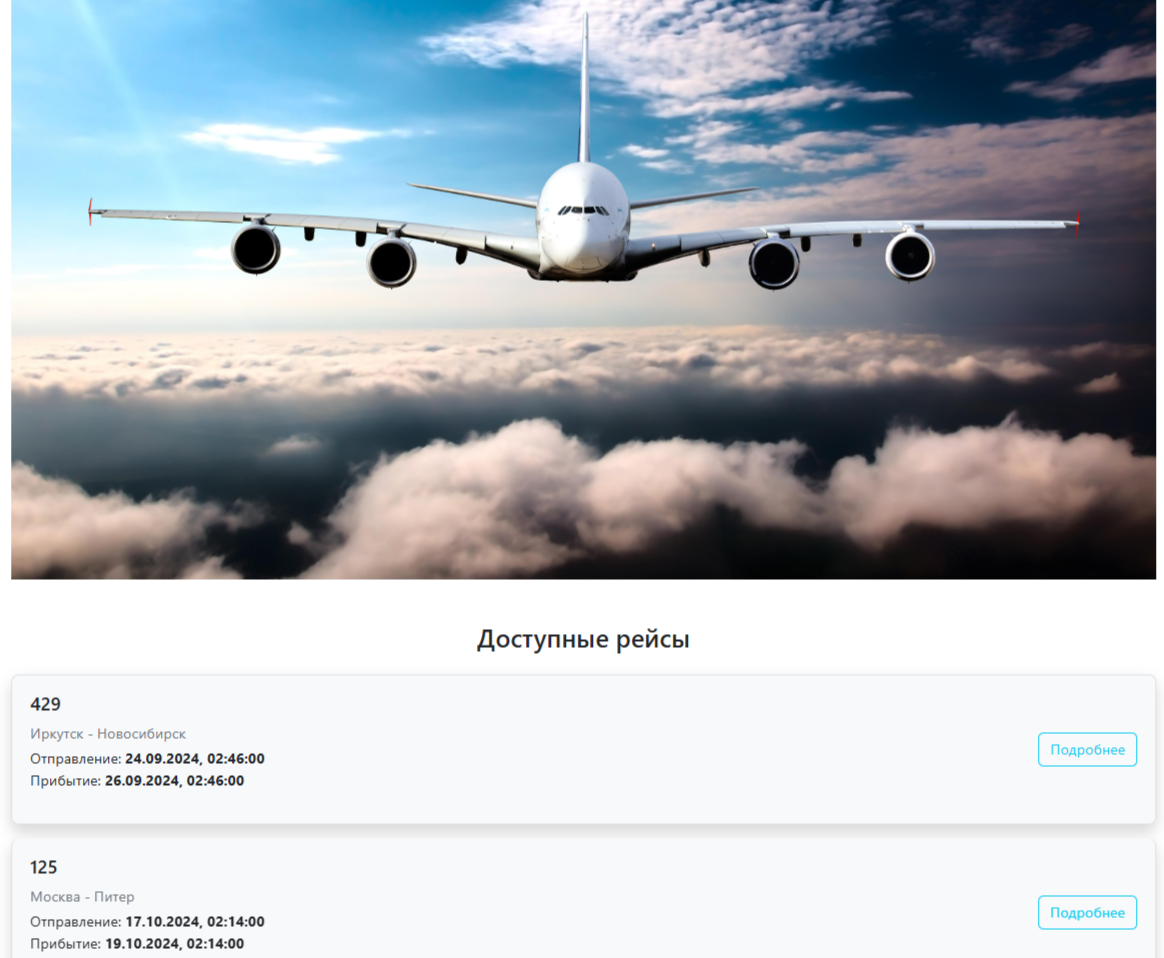


Рисунок 7 – Второй фрагмент страницы «Главная»

Страница «Клиенты»

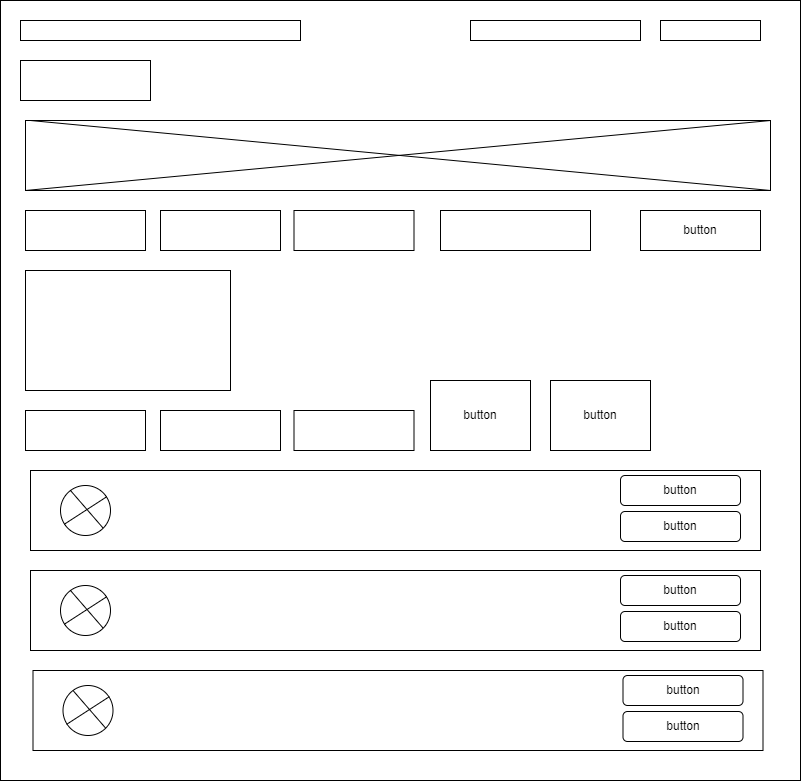


Рисунок 8 – Макет страницы «Клиенты»

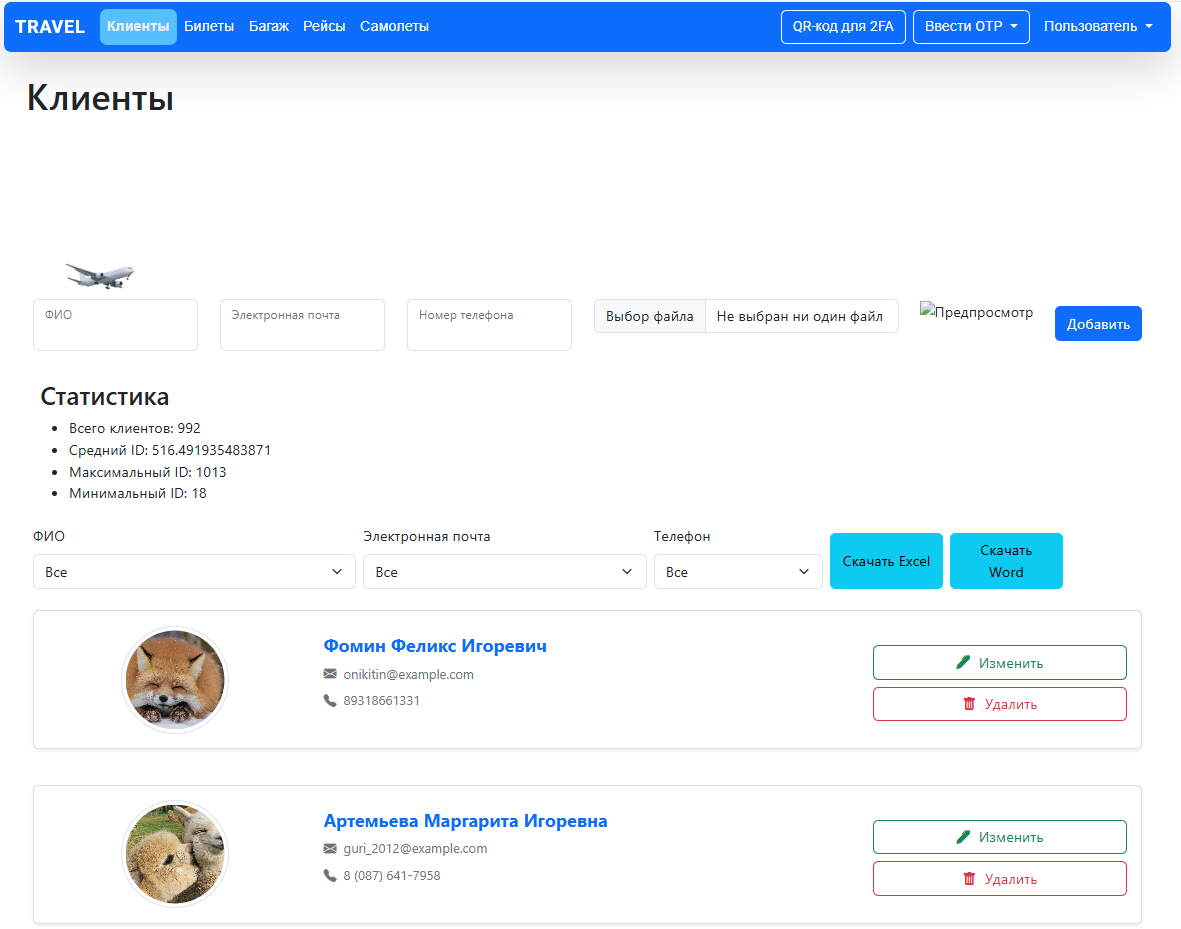


Рисунок 9 – Дизайн страницы «Клиенты»

Страница «Билеты»

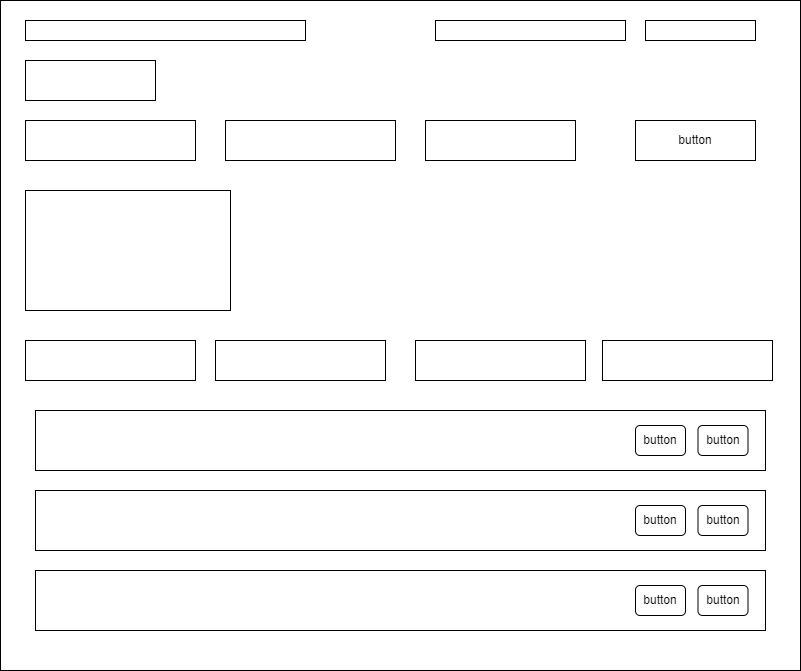


Рисунок 10 – Макет страницы «Билеты»

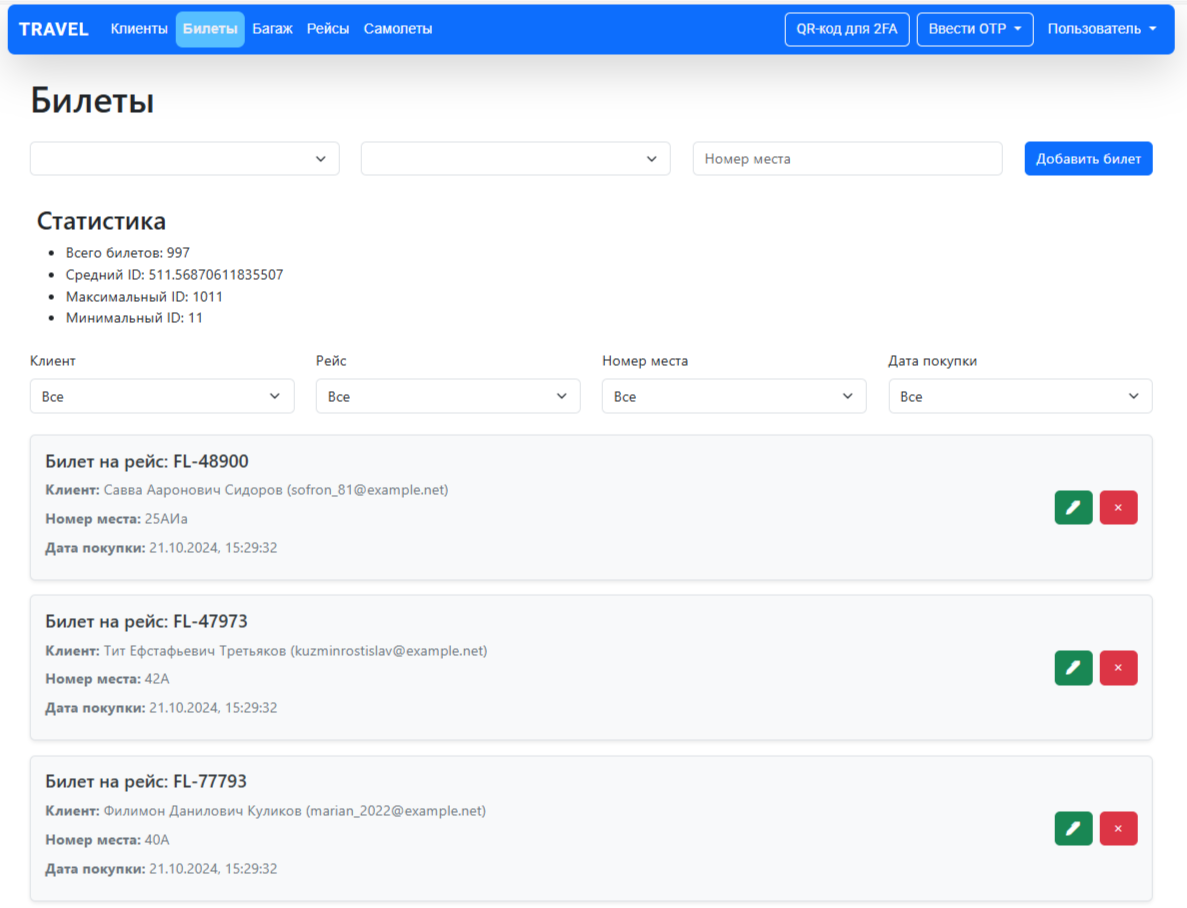


Рисунок 11 – Дизайн страницы «Билеты»

Страница «Багаж»

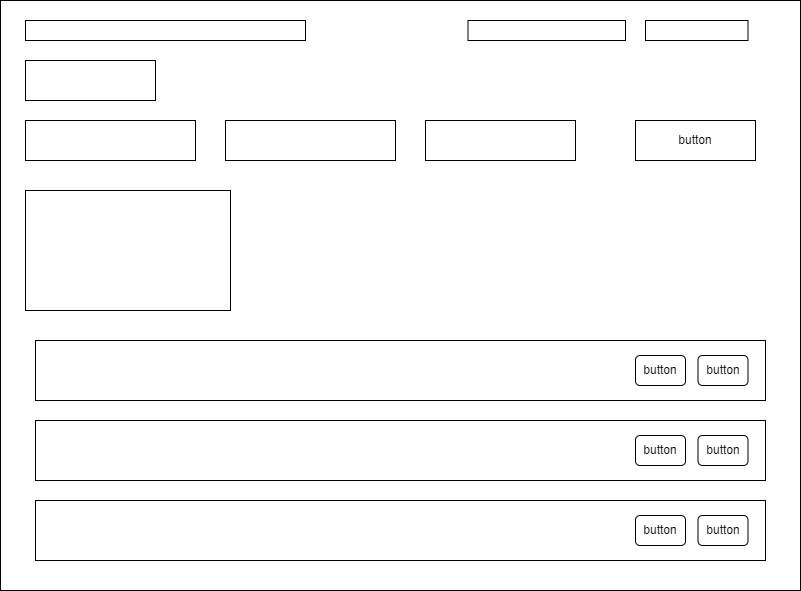


Рисунок 12 – Макет страницы «Багаж»

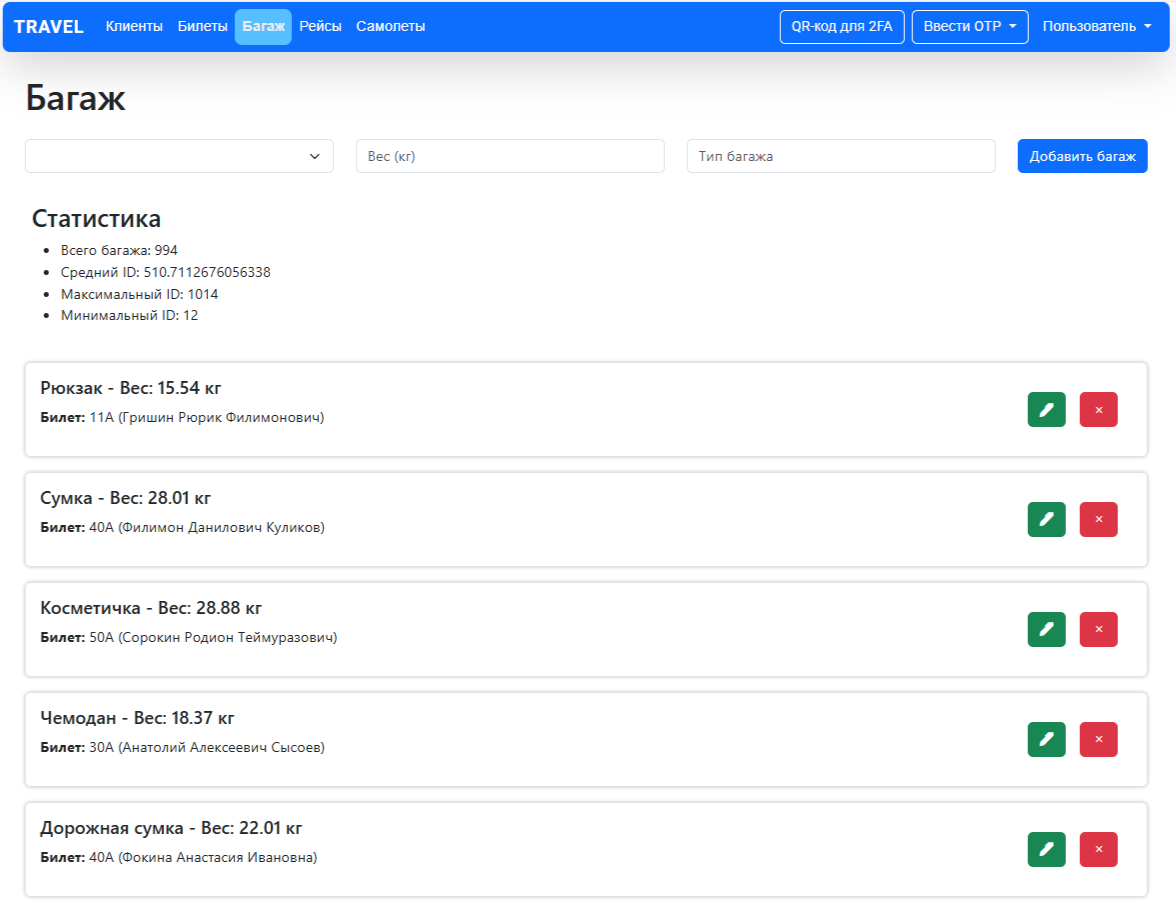


Рисунок 13 – Дизайн страницы «Багаж»

Страница «Рейсы»

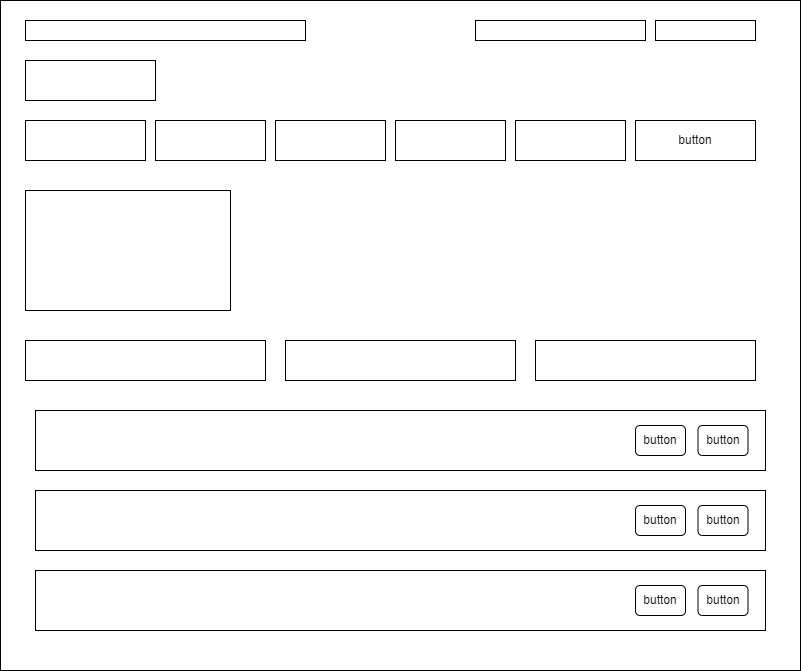


Рисунок 14 – Макет страницы «Рейсы»

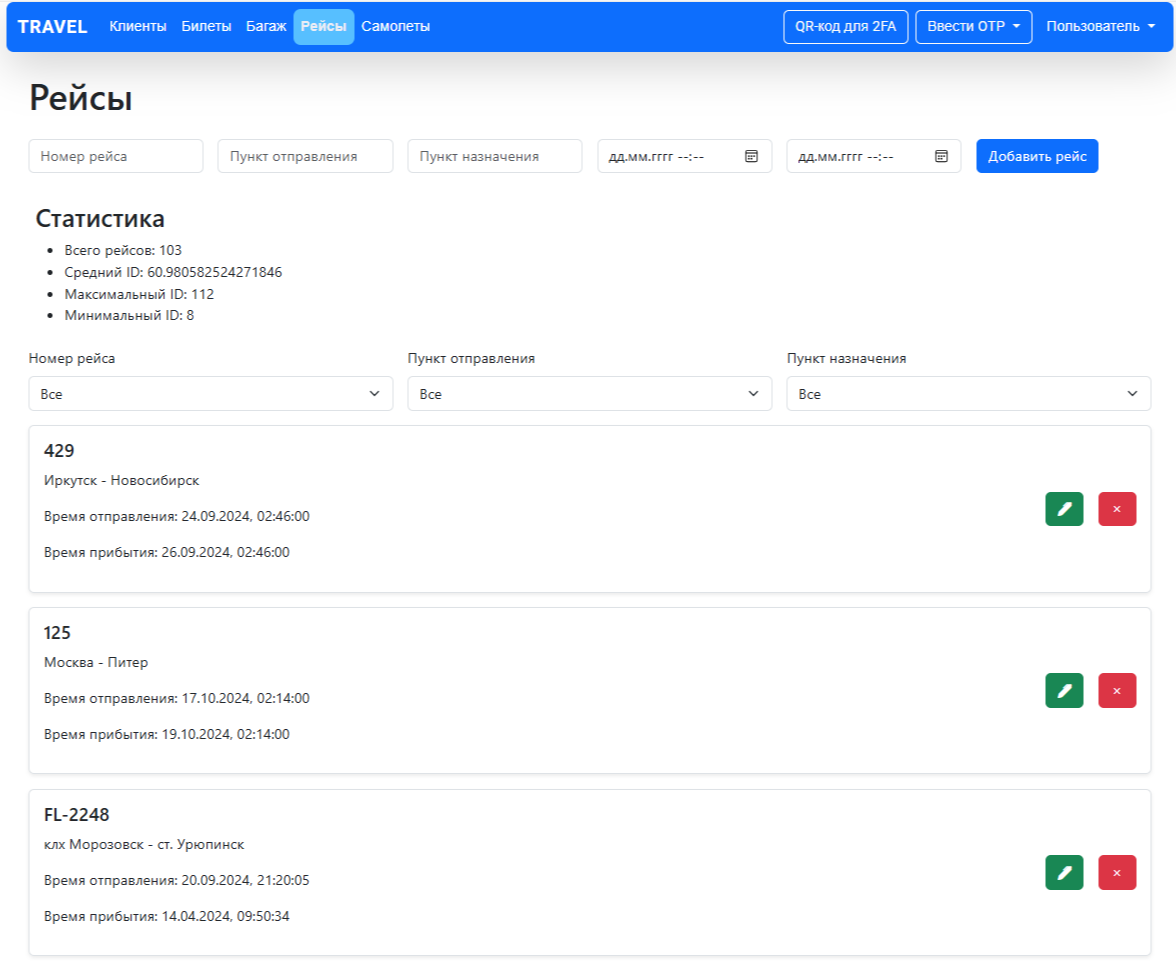


Рисунок 15 – Дизайн страницы «Рейсы»

Страница «Самолеты»

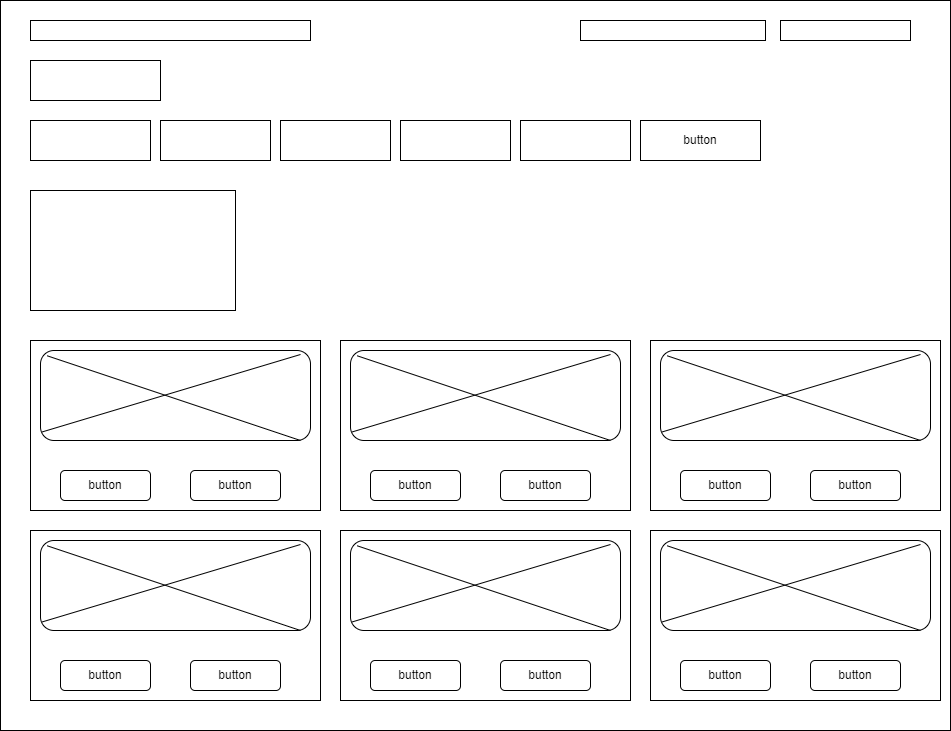


Рисунок 16 – Макет страницы «Самолеты»

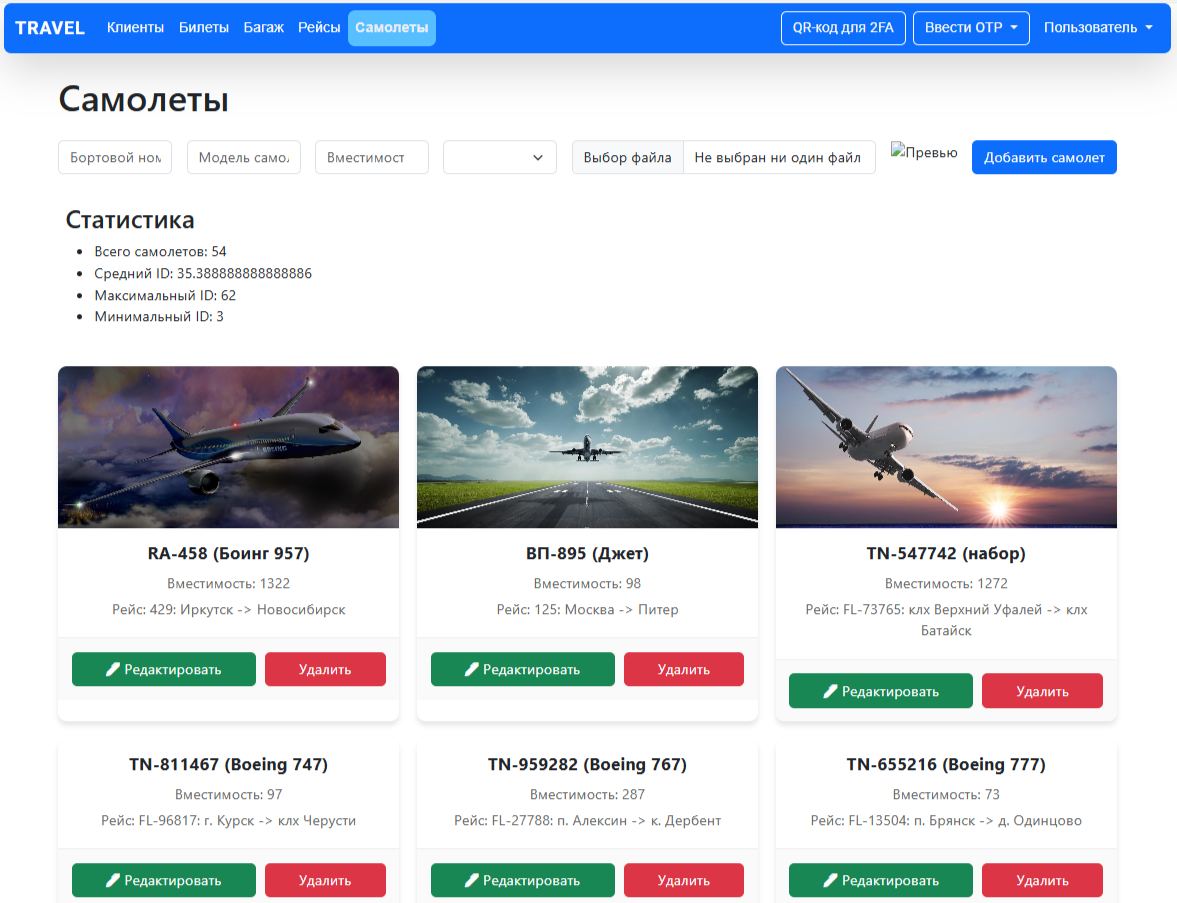


Рисунок 17 – Дизайн страницы «Самолеты»

Страница для входа

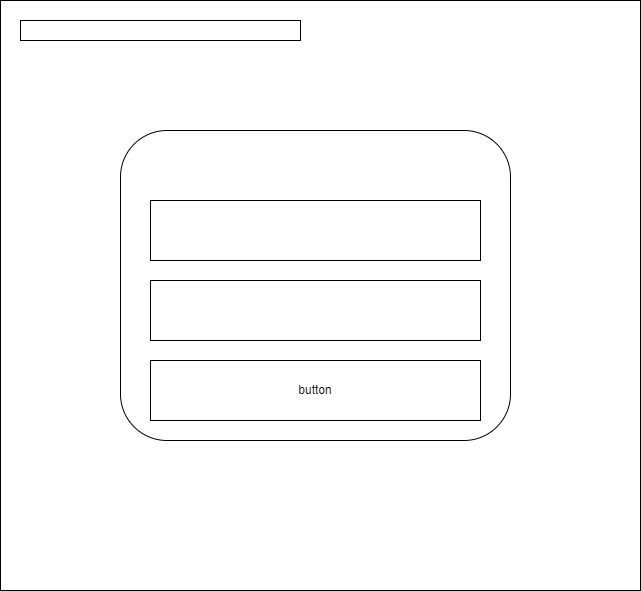


Рисунок 18 – Макет страницы входа

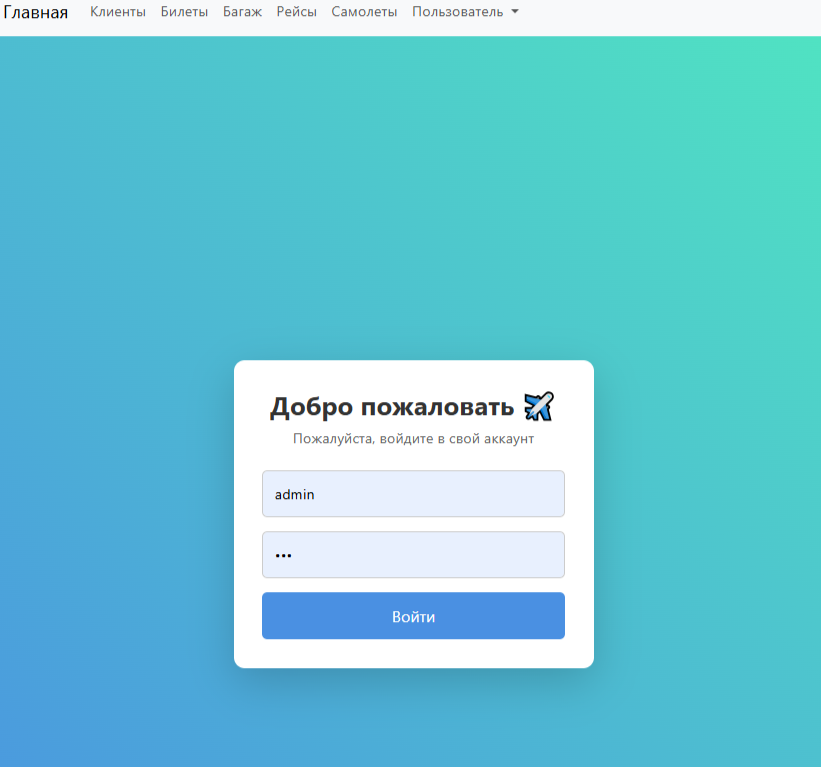


Рисунок 19 – Дизайн страницы входа

# **6 Функции, доступные на сайте**

Пользователь осуществляет вход через страницу личного кабинета, где пользователь вводит логин и пароль, после чего происходит проверка пользователя, если пользователь является суперюзером, то ему впоследствии доступны все записи сайта, если нет, то только те данные, которые были созданы конкретно этим пользователем.

Добавление данных:

Рейсы: возможность добавлять информацию, включая детали о времени отправления, прибытия, номер рейса, пункты отправления и назначения

Багаж: добавление информации, включая его характеристики (вес, тип) и привязка к конкретным пассажирам.

Клиенты: возможность регистрации новых клиентов, включая их ФИО, электронную почту, номер телефона и фото.

Билеты: добавление новых билетов, привязка к рейсам и пассажирам, а также номер места.

Самолеты: добавление данных о новых воздушных судах, их технических характеристиках и назначении на рейсы.

Редактирование данных осуществляется через модальное окно и позволяет изменить любое поле.

Удаление данные осуществляется по кнопке напротив каждой записи.

На страницах «Клиенты», «Билеты» и «Рейсы» есть возможность фильтрации записей по некоторым параметрам.

Удобная навигация между разделами (Главная, Рейсы, Багаж, Клиенты, Билеты, Самолеты) с возможностью переходить к необходимым страницам одним кликом.

# **7 Отличительная особенность**

Реализована интерактивная анимация с изображением самолетика, которая зависит от заполнения формы для добавления нового клиента. Самолетик изменяет свое положение на экране в зависимости от того, какие поля в форме заполнены. Это позволяет пользователю наглядно видеть прогресс в заполнении формы.

Логика анимации:

1. Начальная позиция (самолет на месте): если ни одно из полей формы (ФИО, email, телефон) не заполнено, самолетик остается на стартовой позиции.
2. Взлет: когда пользователь заполняет хотя бы одно из поле ФИО, самолетик начинает "взлетать", то есть он поднимается в верхнюю часть экрана, изображая движение вверх с небольшим наклоном.
3. Полёт: если заполнены два поля (ФИО и email), самолетик перемещается в центральную часть экрана, изображая «полёт».
4. Посадка: когда все обязательные поля формы (ФИО, email, номер телефона) заполнены, самолетик начинает «приземляться» в правой части экрана.

Когда пользователь начинает заполнять форму, срабатывает функция updatePlaneState, которая проверяет состояние каждого поля формы и в зависимости от этого изменяет класс самолетика, что приводит к его перемещению по экрану.

# **Заключение**

В ходе выполнения работы был разработан веб-сайт, предназначенный для удобного и интуитивно понятного доступа к информации о рейсах, багаже, клиентах и других связанных данных в авиационной сфере. Сайт предоставляет эффективные инструменты для управления данными и включает несколько ключевых страниц, каждая из которых выполняет свою функциональную задачу.

Приложение работает с базой данных, что позволяет хранить информацию о билетах, багаже, самолетах клиентах и рейсах. Для взаимодействия между клиентской частью и сервером используется REST API, что обеспечивает эффективную обработку запросов и обновление данных.

В процессе разработки были получены знания работы с фреймворком Vue.js для создания динамичных веб-приложений, а также навыки работы с Axios для осуществления запросов к серверу. Особое внимание было уделено реализации двухфакторной аутентификации и корректной обработке ошибок.

В результате работы получено приложение, которое эффективно справляется с задачами организации и управления данными в области авиаперевозок, и может быть использовано как сотрудниками аэропортов, так и представителями авиакомпаний.

# **Список использованных источников**

1. Django [Электронный ресурс] // Чаинка [сайт]. URL: https://django.tealeaf.su/ (Дата обращения: 01.12.2024)
2. Репозиторий проекта [Электронный ресурс] // Github [сайт]. URL: https://github.com/Veronika-Gerasimova/web\_courseWork