# Документация по работе с сервисом

Кузнецов Максим

Кочура Данил

Коротков Дмитрий

Политех имени Баумана, 2023

# 1 Оглавление

# 1. Функциональная архитектура системы

Сервис состоит из трех разделов:

- Страница для просмотра динамики бронирования
- Страница для просмотра динамики по пролетевшим пассажирам и сезонности
- Страница для прогнозирования спроса на заданный рейс

Все страницы имеют одинаковую структуру  и состоят из трех основных частей:

- Формы для ввода входных данных для анализа
- Графика анализируемой величины
- Скроллбара для масштабирования графика

Графики на всех страницах можно отключить, кликнув на соответствующий блок в легенде графика.

## 1.1. Просмотр динамики бронирования

**Поля формы:**

- Аэропорт вылета
- Аэропорт прилета
- Номер рейса
- Дата вылета рейса
- Класс бронирования

**Все поля являются обязательными**. Номер рейса будет предложен автоматически, исходя из аэропортов направления-назначения.

**Графики:**

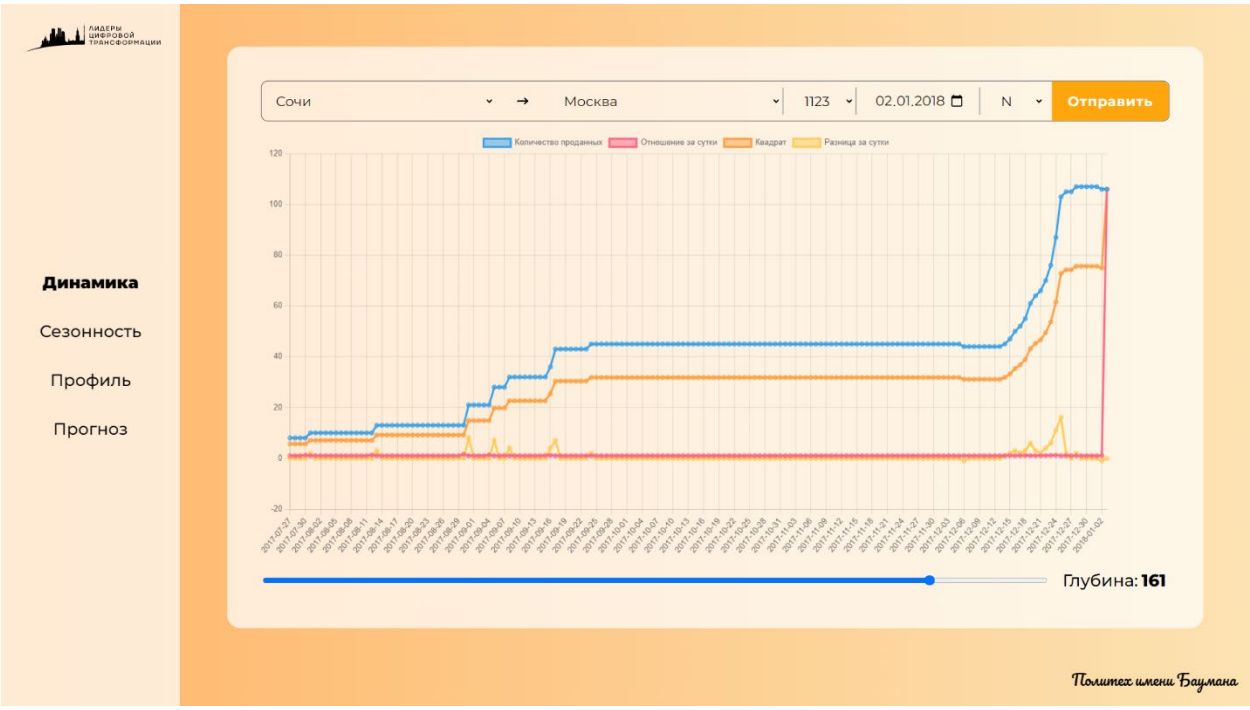
После заполнения формы пользователю будут предоставлены **четыре линейных графика**:

- Количество забронированных мест на рейс за каждый день.
- График отношения количества забронированных мест между соседними днями.
- График среднеквадратичного расстояния между величинами забронированных мест у соседних дней.
- Графи разности количества проданных мест между двумя соседними днями. Фактически – количество забронированных за день мест.

Все величины отложены по оси У, по Х отложены даты. В начале координат – дата открытия рейса к продаже, в конце – дата вылета.

По умолчанию, глубина бронирования максимальна, изменяя  положение ползунка можно масштабировать график в сторону уменьшения глубины бронирования.

**Визуал:**




## 1.2. Просмотр динамики пролетевших пассажиров и сезонности

**Поля формы:**

- Аэропорт вылета
- Аэропорт прилета
- Номер рейса
- Период анализа (год)
- Класс бронирования

**Все поля являются обязательными**. Номер рейса будет предложен автоматически, исходя из аэропортов направления-назначения.

**Графики:**

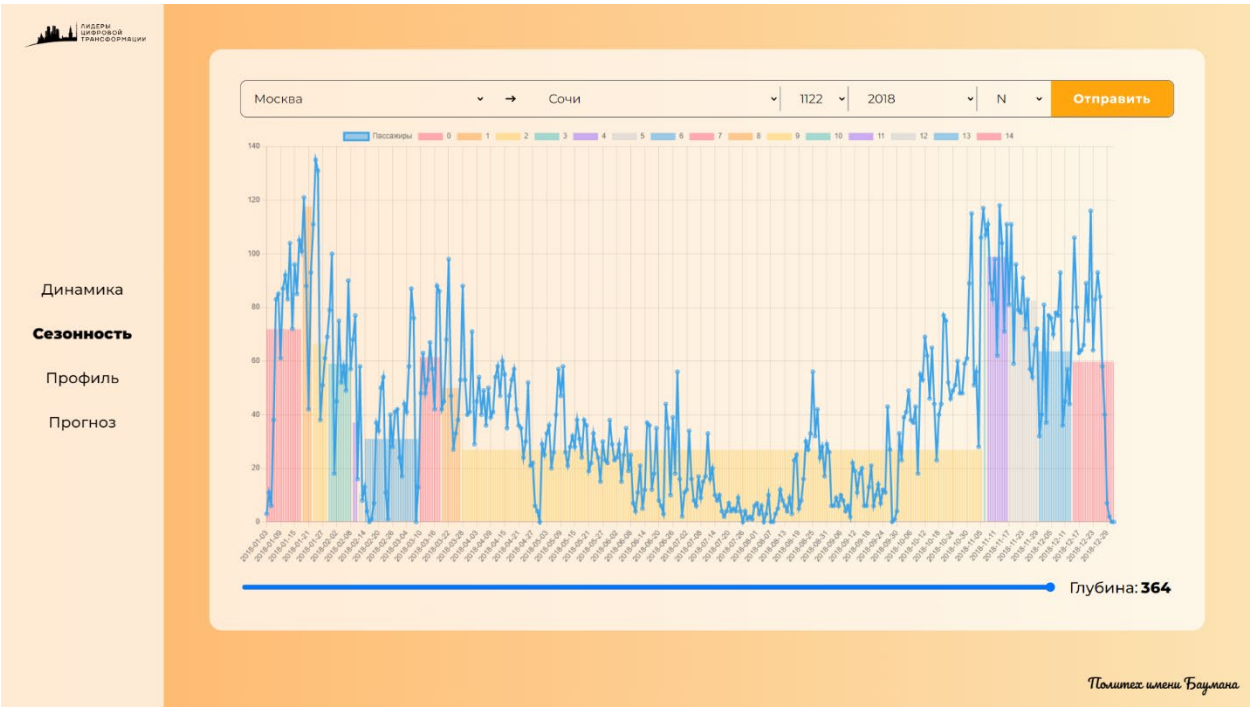
После заполнения формы пользователю будут предоставлены **семейство графиков**:

- Один линейный график динамики пролетевших пассажиров за каждый день
- Несколько одиночных столбчатых диаграмм, соответствующих областям сезонности, определенных алгоритмом

Все величины отложены по оси У, по Х отложены даты. В начале координат – 1 января выбранного года, в конце – 31 декабря.

По умолчанию, глубина бронирования максимальна, изменяя положение ползунка можно масштабировать график в сторону уменьшения глубины в сторону конца года.

**Визуал:**




## 1.3. Прогнозирование спроса на будущие рейсы

**Поля формы:**

- Аэропорт вылета
- Аэропорт прилета
- Номер рейса
- Дата вылета
- Тип ВС
- Класс бронирования

**Все поля являются обязательными**. Номер рейса будет предложен автоматически, исходя из аэропортов направления-назначения. Тип ОС будет предложен автоматически исходя из номера рейса.

**Графики:**

После заполнения формы пользователю будут предоставлен **один линейный график**:

- Прогноз спроса на выбранный рейс на всей глубине бронирования.

Величина прогноза – предполагаемое количество покупателей за каждый день отложено по оси У, по Х отложено количество дней до закрытия рейса. В начале координат – максимальная доступная глубина, в конце – 0 (дата вылета)

По умолчанию, глубина бронирования максимальна, изменяя положение ползунка можно масштабировать график в сторону уменьшения глубины в сторону даты вылета

**Визуал:**


## 1.4. Заключение

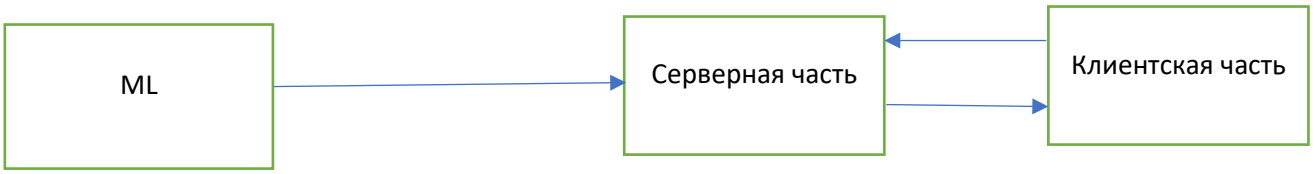
Перемещение между тремя страницами организовано через меню в левой части страницы.

Интерфейс понятен и прост в использовании, реализованы дополнения, облегчающие работу с сервисом .

# 2. Программно-техническая архитектура системы

## 2.1. Структура проекта

Проект состоит из двух основных частей – заранее обученной ML-модели и веб-сервиса




Веб-сервис можно условно разделить на серверную и клиентскую часть.

**Серверная** часть реализована на стеке:

- PHP (7.4>=) – основная бизнес-логика приложения и API
- MySQL – база данных

**Клиентская** часть, помимо самописных компонентов на CSS и JavaScript использует следующие сторонние библиотеки:

- jQuery https://jquery.com/
- ChartJS https://www.chartjs.org

Модель реализована на языке Python и обучена средствами следующих программных пакетов:

- NumPy https://numpy.org/
- Pandas https://pandas.pydata.org/
- Catboost https://catboost.ai/

Скрипт, который отвечает за прогнозирование написан на Python с использованием упомянуых выше библиотек. Взаимодействует с серверной частью приложения через консоль ОС, где запускается из-под интерпретатора PHP.

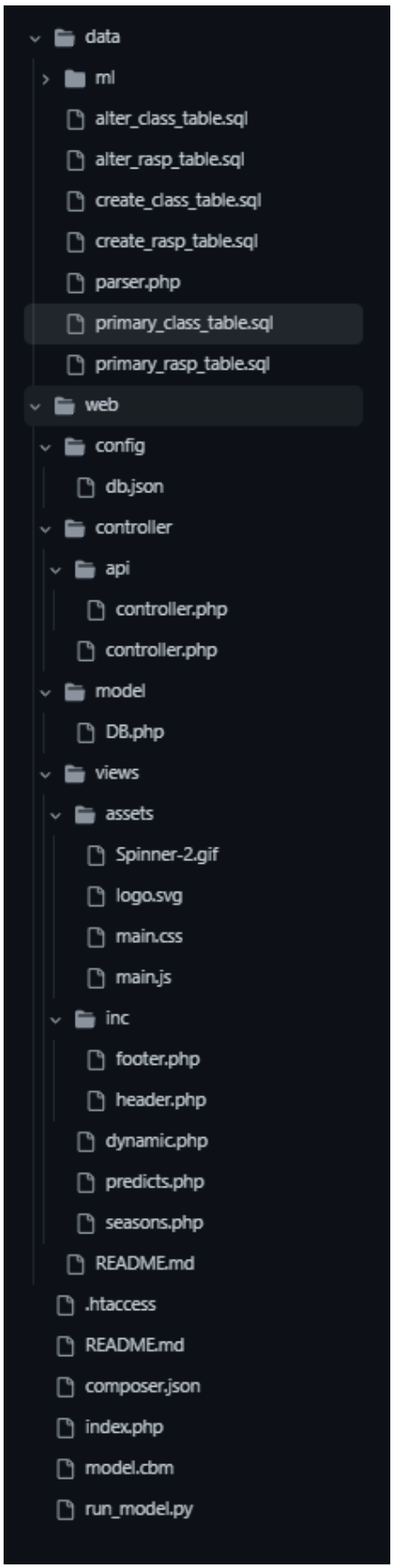
## 2.1. Архитектура веб-приложения

Файл Index.php в корне проекта – единая точка входа в веб-приложение. Фйлы .htaccess задает настройки Apache-сервера для перенаправления всех запросов на этот файл.

Структура серверной части реализует очень примитивную версию  MVC-паттерна:

В зависимости от запроса index.php подключает один из контролеров (/web/controller/), которые обрабатывают запрос, получает данные и отправляют ответ.

Файловая структура выглядит следующим образом:




- data/ – Директория для работы с бд и исходными данными
- Файлы data/*.sql – заготовленная структура базы данных
- data/parser.php – скрипт для парсинга данных из csv-таблиц в БД

- web/config/config.json – файл с настройками конфигурации БД
- controller/controller.php – контроллер-маршрутизатор для клиентских запросов на страницы
- controller/api/controller.php -контроллер-маршрутизатор для API

- model/DB.php – класс для упрощения работы с БД.

- Views/ - директория для хранения клиентских файлов – шаблонов страниц, стилей, скриптов и изображений.

- Views/assets - директория для хранения стилей, скриптов и изображений

- Views/inc/ - директория для хранения переиспользуемых частей интерфейса

- Остальные файлы директории views/ - HTML - шаблоны страниц

- Model.cmb – файл обученной ML-модели
- Run_model.py – скрипт для работы с моделью

# 3. Документация с возможными инцидентами эксплуатации системы.

## 3.1. Инциденты с моделью:

Так как модель недоучена, то она может выдавать некорректные результаты. Из-за того, что не удалось обучить модель с классами бронирования, мы учитываем только салоны.

## 3.2. Инциденты с работой веб-сервиса:

В клиентской части приложения реализована обработка исключений с API и Бэкенда. В случае возникновения ошибок всплывает окно с сообщением о некорректно введенных данных, отсутствии информации и т.д.