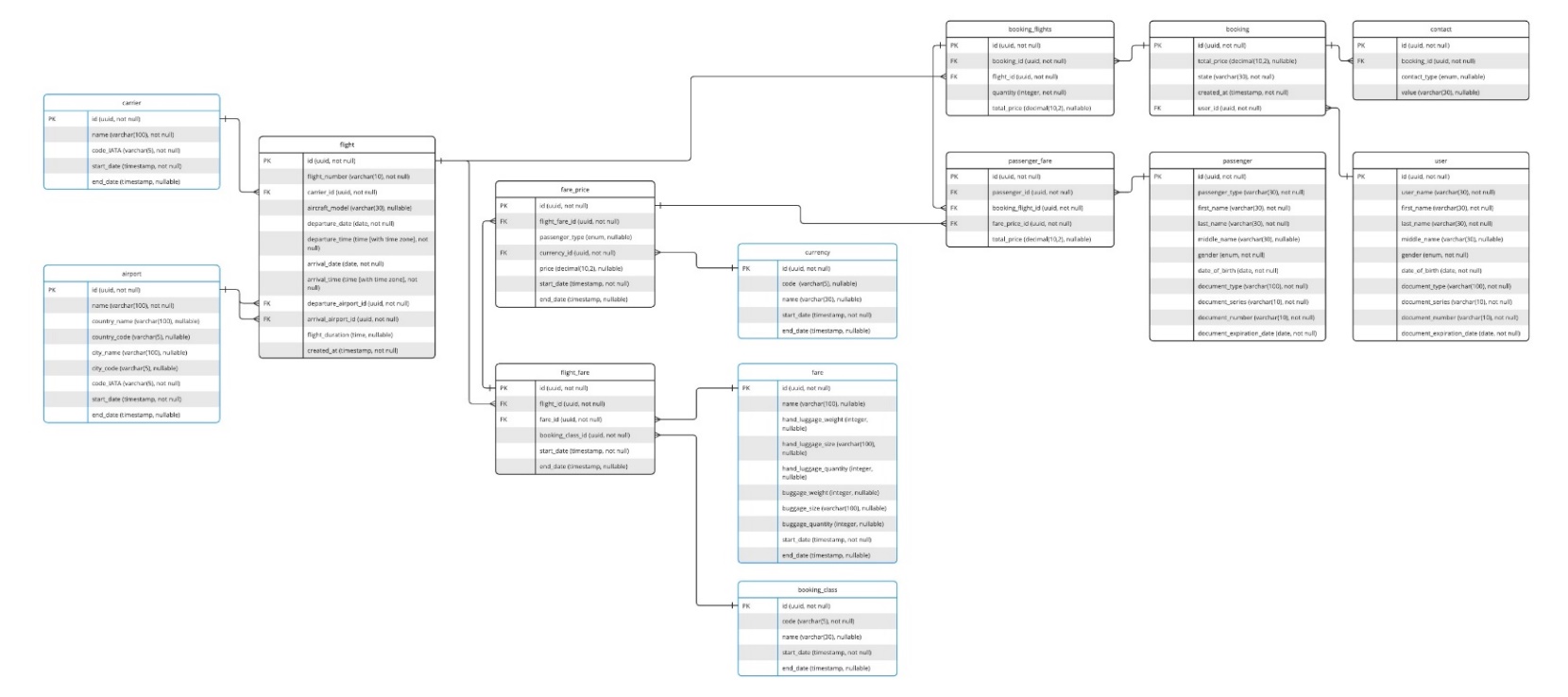
Бронирование авиабилетов



Пользовательские требования в формате User Story

Как пользователь сайта, я хочу получать список доступных рейсов с ценами по фильтрам "Город назначения". "Город прибытия", "Дата отправления", чтобы находить нужные мне рейсы на конкретную дату.

Как пользователь сайта, я хочу фильтровать найденные рейсы по наличию багажа и выполнять сортировку по стоимости, чтобы находить самое выгодное предложение.

Как пользователь сайта, я хочу иметь возможность оформить бронирование на несколько пассажиров, чтобы я мог купить билеты на один рейс вместе со своими друзьями и семьей.

Как пользователь сайта, я хочу просматривать список своих предстоящих и прошедших полетов.

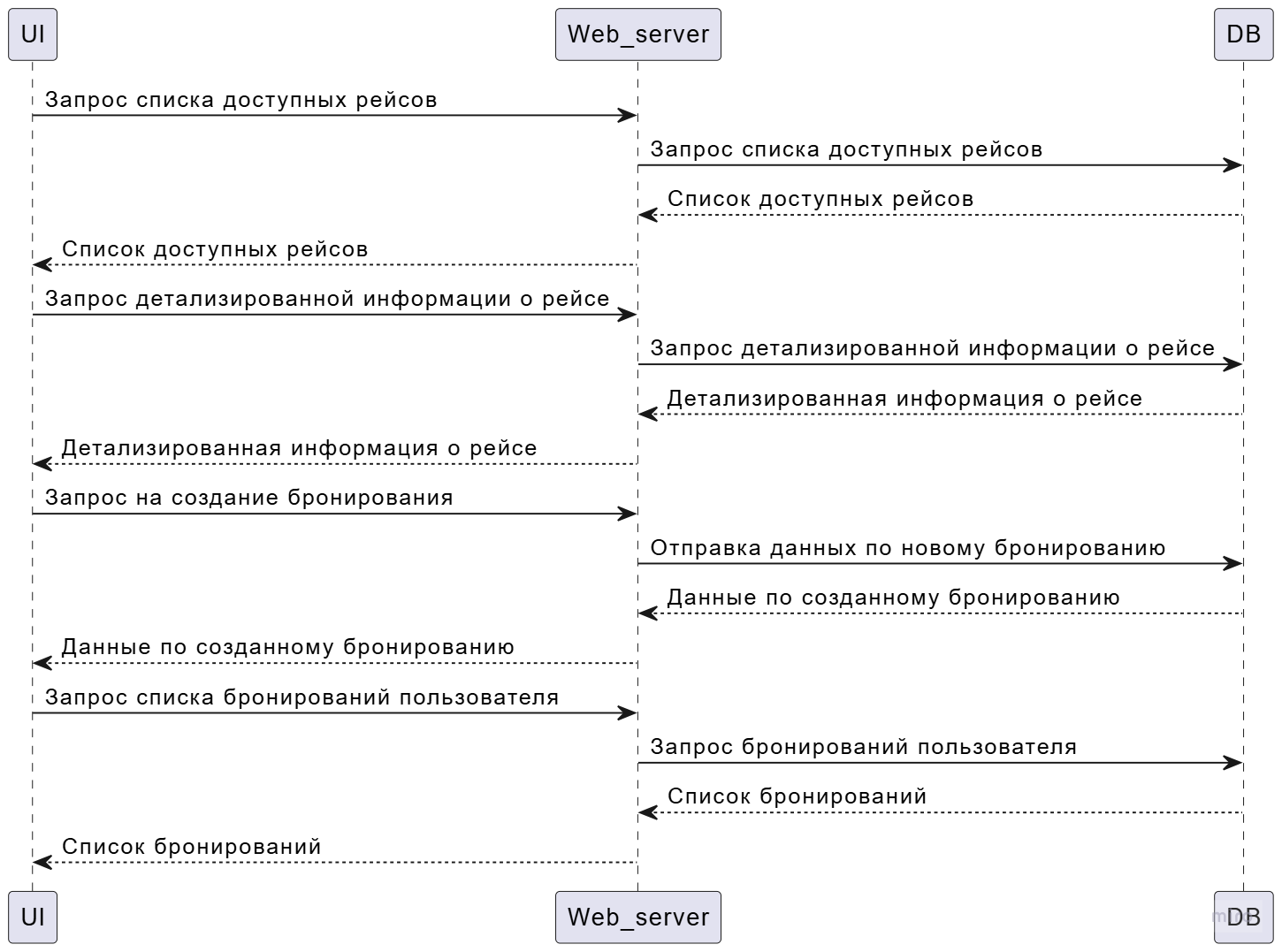
На основании User Story можно выделить следующие сценарии использования:

получение списка рейсов по фильтрам пользователя (локация вылета и прилета, дата, количество пассажиров, наличие багажа и сортировка по стоимости);

получение детализированной информации по выбранному рейсу для бронирования;

создание нового бронирования;

получение списка бронирований пользователя.



Мы пойдем по такому пути:

1)определим endpoints нашего ApI;

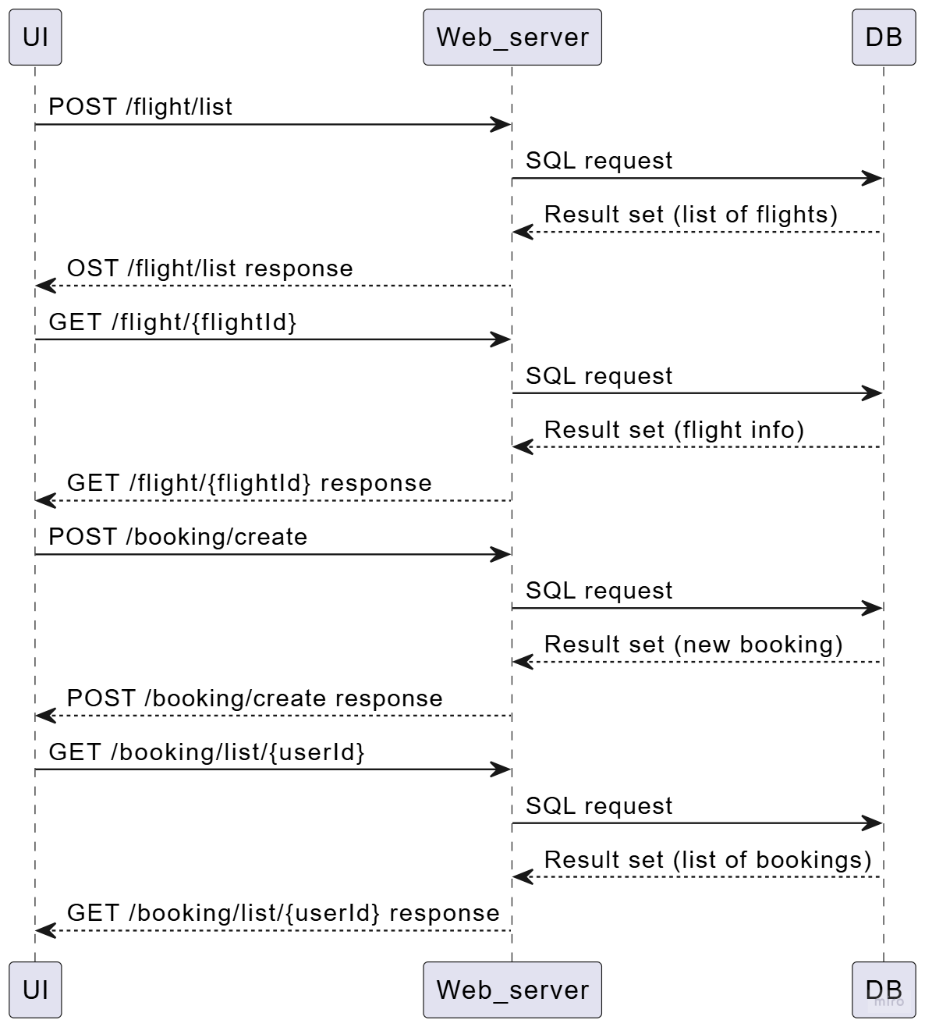
2)определим модели данных (схемы), которые будут использоваться в ApI

3)определим параметры и возможные ответы для каждого endpoint;

4)на основании полученной структуры данных оформим OpenApI документацию в формате YAML, используя интерфейс SwaggerHub, который поддерживает встроенную валидацию кода.

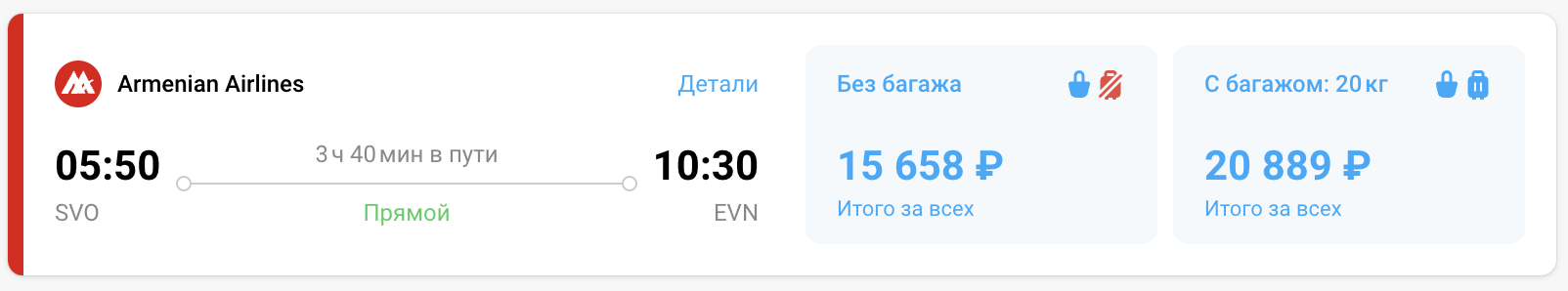
|  |  |
| --- | --- |
| pOST /**flight**/list - получение списка рейсов по фильтрам пользователя;  GET /**flight**/{flightId} - получение информации по рейсу по id рейса;  pOST /**booking**/create - создание нового бронирования;  GET /**booking**/list/{userId} - получение бронирований пользователя по его id. | получение списка рейсов по фильтрам пользователя (локация вылета и прилета, дата, количество пассажиров, наличие багажа и сортировка по стоимости);  получение детализированной информации по выбранному рейсу для бронирования;  создание нового бронирования;  получение списка бронирований пользователя. |

# 1) Определим endpoints нашего ApI (список запросов)



# 2) Определим модели данных (схемы), которые будут использоваться в ApI

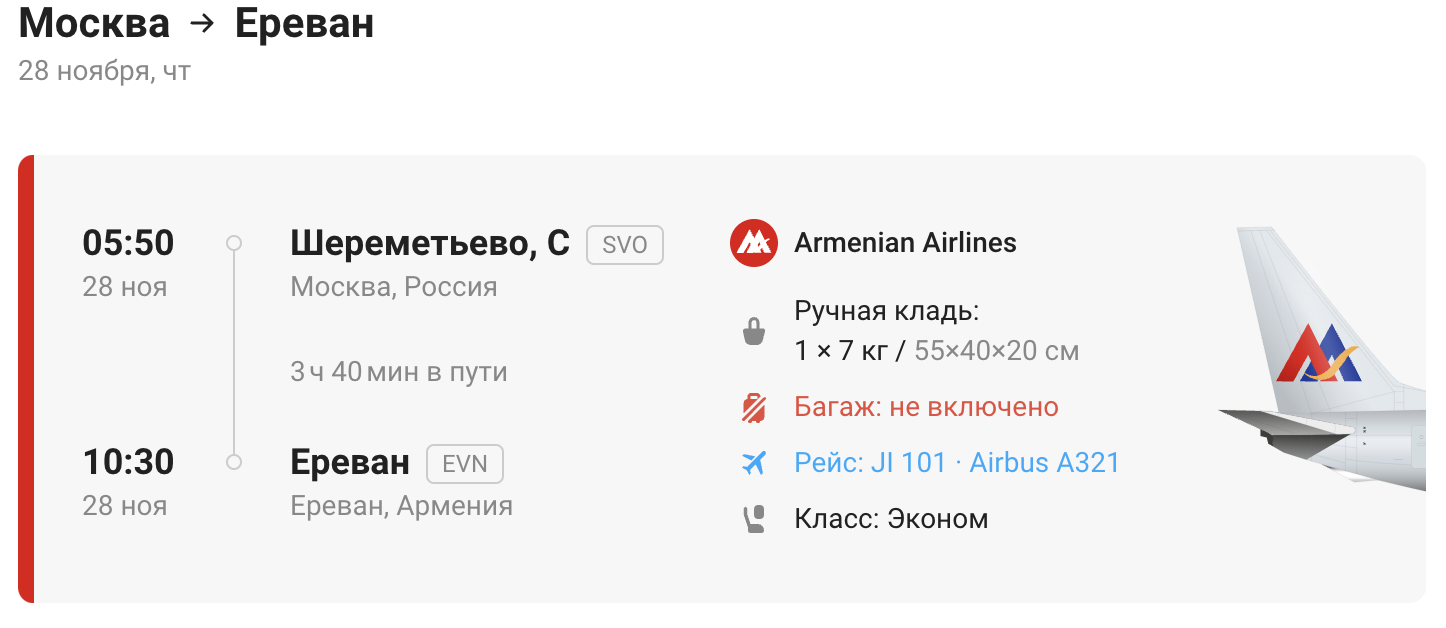
2.1) Построии JSON структуру dto “**flight**



Мы **ожидаем увидеть** на форме следующую информацию по рейсу:

* авиакомпанию;
* аэропорт и время отправления;
* аэропорт и время прибытия;
* время в пути;
* краткую информацию по тарифу: наличие и вес багажа, стоимость за всех пассажиров.

**Также мы помним, что нам требуется фильтрация по наличию багажа и сортировка по стоимости**.



Итак какю информацию **мы ожидаем** увидеть вместе со всеми фильтрами

* авиакомпанию;
* аэропорт, терминал, дату и время отправления;
* аэропорт, терминал, дату и время  прибытия;
* время в пути;
* номер рейса и модель самолета;
* класс обслуживания;
* подробную информацию по тарифу:
  + количество, вес и габариты ручной клади;
  + наличие, количество и вес багажа (если тариф с багажом).

dto “**flight”**

{

"id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",

"flightNumber": "0524",

"carrier": {

"id": "e13259d5-7c73-4fb2-8b86-38408d30202c",

"name": "Аэрофлот",

"codeIATA": "SU"

},

"aircraftModel": "Airbus A321",

"departureDate": "2024-10-10",

"departureTime": "10:15:00",

"arrivalDate": "2024-10-10",

"arrivalTime": "11:45:00",

"flightDuration": "1:30",

"departureAirport": {

"id": "acd11a35-9768-4228-9ff6-e66929e4dd8a",

"name": "Шереметьево",

"codeIATA": "SVO",

"countryName": "Россия",

"countryCode": "RU",

"cityName": "Москва",

"cityCode": "MOW",

},

"arrivalAirport": {

"id": "6811f72c-5dab-4156-91d8-ff1ac8a575c2",

"name": "Пулково",

"codeIATA": "LED"

"countryName": "Россия",

"countryCode": " RU",

"cityName": "Санкт-Петербург",

"cityCode": "LED",

},

"fares": [

{

"id": "ada59145-c4bd-4cc8-8c6c-3d09b0d5153f",

"": {

"id": "8a5a56d1-807b-48a7-a899-b2d56b6be730",

"code": "QCL",

"name": "Эконом"

}

"name": "С багажом",

"handLuggageWeight": 10,

"handLuggageSize": "55x40x25",

"handLuggageQuantity": 1,

"buggageWeight": 23,

"buggageSize": "203",

"buggageQuantity": 1,

"prices": [

{

"id": "8e60c4d6-0a38-4359-8e97-99ca6fe88bb1",

"passengerType": "ADULT",

"price": "8000.00",

"currency": {

"id": "29b51335-2473-4739-a6f5-d29eed6fadd2",

"code": "RUB",

"name": "Российский рубль"

}

}

]

}

}

]

Но мы видим, что он включает в себя объекты и массивы объектов. Для них нам также потребуются отдельные dto:

carrier: запись справочника Авиакомпания;

currency: запись справочника Валюта;

bookingClass: запись справочника Класс обслуживания;

Airport: запись справочника Аэропорт;

fare: запись справочника Тариф;

fares-flightfareprice: данные о ценах на тарифы на конкретном рейсе.

prices- Price: структура данных с ценами на тариф;

|  |  |  |
| --- | --- | --- |
| dto json | dto yaml | DTO YAML |
| airport: запись справочника аэропорт;  "airport": {  "id": "6811f72c-5dab-4156-91d8-ff1ac8a575c2",  "name": "пулково",  "codeiata": "led"  "countryname": "россия",  "countrycode": " ru",  "cityname": "санкт-петербург",  "citycode": "led"  }  carrier: запись справочника авиакомпания  "carrier": {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "name": "аэрофлот",  "codeiata": "su"  }  currency: запись справочника валюта  " currency ": {  "id": "b32f6142-8df9-4a30-980b-45ffb391d5b9",  "code": "usd",  "name": "доллар сша"  }  flight: данные по рейсу  {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "flightnumber": "0524",  "carrier": {  "id": "e13259d5-7c73-4fb2-8b86-38408d30202c",  "name": "аэрофлот",  "codeiata": "su"  },  "aircraftmodel": "airbus a321",  "departuredate": "2024-10-10",  "departuretime": "10:15:00",  "arrivaldate": "2024-10-10",  "arrivaltime": "11:45:00",  "flightduration": "1:30",  "departureairport": {  "id": "acd11a35-9768-4228-9ff6-e66929e4dd8a",  "name": "шереметьево",  "codeiata": "svo",  "countryname": "россия",  "countrycode": "ru",  "cityname": "москва",  "citycode": "mow",  },  "arrivalairport": {  "id": "6811f72c-5dab-4156-91d8-ff1ac8a575c2",  "name": "пулково",  "codeiata": "led"  "countryname": "россия",  "countrycode": " ru",  "cityname": "санкт-петербург",  "citycode": "led",  },  "fares": [  {  "id": "ada59145-c4bd-4cc8-8c6c-3d09b0d5153f",  "": {  "id": "8a5a56d1-807b-48a7-a899-b2d56b6be730",  "code": "qcl",  "name": "эконом"  }  "name": "с багажом",  "handluggageweight": 10,  "handluggagesize": "55x40x25",  "handluggagequantity": 1,  "buggageweight": 23,  "buggagesize": "203",  "buggagequantity": 1,  "prices": [  {  "id": "8e60c4d6-0a38-4359-8e97-99ca6fe88bb1",  "passengertype": "adult",  "price": "8000.00",  "currency": {  "id": "29b51335-2473-4739-a6f5-d29eed6fadd2",  "code": "rub",  "name": "российский рубль"  }  }  ]  }  }  ]  fare: запись справочника тариф  "fare": {  "name": "с багажом",  "handluggageweight": 10,  "handluggagesize": "55x40x25",  "handluggagequantity": 1,  "buggageweight": 23,  "buggagesize": "203",  "buggagequantity": 1  }  price: структура данных с ценами на тариф  “price”:{  "id": "aa0d3d12-53df-4d0a-b8e5-1d2f3ef3f9fa",  "passengertype": "adult",  "price": 12000.50,  "currency": {  "id": "b32f6142-8df9-4a30-980b-45ffb391d5b9",  "code": "rub",  "name": "российский рубль"  }  }  flightfareprice: данные о ценах на тарифы на конкретном рейсе  “flightfareprice“:{  "id": "13fb7991-4269-43c6-9f5e-099d21e6d893",  "fare": {  “name": "с багажом",  "handluggageweight": 10,  "handluggagesize": "55x40x25",  "handluggagequantity": 1,  "buggageweight": 23,  "buggagesize": "203",  "buggagequantity": 1  },  " bookingclass ": {  "id": "c92e637e-3f6b-4b50-a646-b6ec4e9e2172",  "code": "bus",  "name": "бизнес-класс"  },  “price”:{  "id": "aa0d3d12-53df-4d0a-b8e5-1d2f3ef3f9fa",  "passengertype": "adult",  "price": 12000.50,  "currency": {  "id": "b32f6142-8df9-4a30-980b-45ffb391d5b9",  "code": "rub",  "name": "российский рубль"  }  }  }, | carrier: #наименование схемы  required: #список обязательных атрибутов  - id  - name  - code  properties: #список атрибутов схемы  id: #наименование атрибута  type: string #тип данных  format: uuid #формат данных  example: 02d837dc-9d8d-4c64-955a-8f0e16df9276 #пример значения  name:  type: string  example: аэрофлот  code:  type: string  example: su  type: object #тип данных корневого элемента    airport:  required:  - id  - name  - codeiata  properties:  id:  type: string  format: uuid  example: 6811f72c-5dab-4156-91d8-ff1ac8a575c2  name:  type: string  example: пулково  codeiata:  type: string  example: led  countryname:  type: string  example: россия  countrycode:  type: string  example: ru  cityname:  type: string  example: санкт-петербург  citycode:  type: string  example: led  type: object   currency:        required:          - id          - code        properties:          id:            type: string            format: uuid            example: 02d837dc-9d8d-4c64-955a-8f0e16df9276          name:            type: string            example: российский рубль          code:            type: string            example: rub        type: object  flight:        required:          - id          - flightnumber          - carrier          - departuretime          - departuredate          - arrivaltime          - arrivaldate          - departureairport          - arrivalairport          - createdat        properties:          id:            type: string            format: uuid            example: 02d837dc-9d8d-4c64-955a-8f0e16df9276          flightnumber:            type: string            example: 0524          carrier:            $ref: '#/components/schemas/carrier'          departureairport:            $ref: '#/components/schemas/airport'          arrivalairport:            $ref: '#/components/schemas/airport'          aircraftmodel:            type: string            example: airbus a321          departuredate:            type: string            format: date            example: '2024-10-10'          departuretime:            type: string            format: time            example: '10:15:00'          arrivaldate:            type: string            format: date            example: '2024-10-10'          arrivaltime:            type: string            format: time            example: '11:45:00'          flightduration:            type: string            example: 1:30:00          fares:            type: array            items:               $ref: '#/components/schemas/flightfareprice'          createdat:            type: string            format: datetime            example: '2024-10-10t11:03:15.838z'        type: object  fare:        required:          - id        properties:          id:            type: string            format: uuid            example: 02d837dc-9d8d-4c64-955a-8f0e16df9276          name:            type: string            example: с багажом          handluggageweight:            type: number            format: "int64"            example: 10          handluggagesize:            type: string            example: 55x40x25          handluggagequantity:            type: number            format: "int64"            example: 1          buggageweight:            type: number            format: "int64"            example: 23          buggagesize:            type: string            example: 203          buggagequantity:            type: number            format: "int64"            example: 1        type: object  price:        required:          - id          - passengertype        properties:          id:            type: string            format: uuid            example: 02d837dc-9d8d-4c64-955a-8f0e16df9276          passengertype:            type: string            enum: [adult, child, infant]            example: adult          price:            type: number            format: digits            example: 8000.00          currency:            $ref: '#/components/schemas/currency'        type: object    flightfareprice:        required:          - id          - price          - fare        properties:          id:            type: string            format: uuid            example: 02d837dc-9d8d-4c64-955a-8f0e16df9276          fare:            $ref: '#/components/schemas/fare'          bookingclass:            $ref: '#/components/schemas/bookingclass'          price:            type: array            items:               $ref: '#/components/schemas/price'        type: object |  |

Выполним такой же анализ для действий с бронированиями.

Получаем JSON-структуру Booking:

{

"id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",

"totalPrice": "8000.00",

"state": "ACTIVE",

"createdAt": "2024-10-10T11:03:15.838Z",

"userId": "02d837dc-9d8d-4c64-955a-8f0e16df9276",

"contacts": [

{

"id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",

"contactType": "PHONE",

"value": "+79999999999"

},

{

"id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",

"contactType": "EMAIL",

"value": "test@test.ru"

}

],

"flights1": [

{

"id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",

"flightNumber": "0524",

"quantity": 1,

"totalPrice": "8000.00"

"passengers": [

{

"id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",

"passengerType": "ADULT",

"firstName": "Райан",

"lastName": "Гослинг",

"middleName": "Иванович",

"gender": "MALE",

"dateOfBirth": "1980-11-12",

"documentType": "RUSSIAN\_PASSPORT",

"documentSeries": "77",

"documentNumber": "123456",

"documentExpirationDate": "2026-01-01",

"fare1": {

"farePriceId": "02d837dc-9d8d-4c64-955a-8f0e16df9276",

"totalPrice": "8000.00",

“currency”: "RUB"

}

}

]

}

]

}

Итого, нам потребуются dto:

* Carrier: запись справочника Авиакомпания;
* Currency: запись справочника Валюта;
* BookingClass: запись справочника Класс обслуживания;
* Airport: запись справочника Аэропорт;
* Fare: запись справочника Тариф;
* Price: структура данных с ценами на тариф;
* FlightFarePrice: данные о ценах на тарифы на конкретном рейсе;
* Flight: данные по рейсу;
* FlightsSearch: набор фильтров для поиска рейсов (к нему мы еще вернемся);
* Passenger: данные по пассажиру;
* Contact: данные таблицы с контактными данными для бронирования;
* fare1-SelectedFarePrice: данные по стоимости тарифа, добавленного в бронирование;
* flights1-PassengerFarePrice: срез данных по пассажиру, выбранному для него тарифу и его стоимости;
* Booking: данные по бронированию.

|  |  |
| --- | --- |
| dto json | dto yaml |
| bookingclass: запись справочника класс обслуживания  “booking\_class ": {  "id": "c92e637e-3f6b-4b50-a646-b6ec4e9e2172",  "code": "bus",  "name": "бизнес-класс"  }  flightssearch: набор фильтров для поиска рейсов (к нему мы еще вернемся);  passenger: данные по пассажиру  passenger”:{  "id": "e123f72c-5dab-4156-91d8-ff1ac8a575c2",  "passengertype": "adult",  "firstname": "иван",  "lastname": "петров",  "middlename": "алексеевич",  "gender": "male",  "dateofbirth": "1985-06-01",  "documenttype": "russian\_passport",  "documentseries": "4510",  "documentnumber": "123456",  "documentexpirationdate": "2030-06-01"  }  contact: данные таблицы с контактными данными для бронирования;  “contact”:{  "id": "f9a6c12b-3d4e-4f9a-b123-4567abcd8901",  "contacttype": "phone",  "value": "+7 912 345 67 89"  }  selectedfareprice: данные по стоимости тарифа, добавленного в бронирование;  “selectedfareprice”:{  "farepriceid": "3e4f5d6a-7c89-4b0d-9e2a-1a2b3c4d5e6f",  "totalprice": 15500,  "currency": {  "id": "b32f6142-8df9-4a30-980b-45ffb391d5b9",  "code": "usd",  "name": "доллар сша"  }  }  passengerfareprice: срез данных по пассажиру, выбранному для него тарифу и его стоимости;  “passengerfareprice”:{  "id": "4b9f0d7a-81c2-44d3-a9f7-1234567890ab",  "passengertype": "adult",  "firstname": "алексей",  "lastname": "иванов",  "middlename": "сергеевич",  "gender": "male",  "dateofbirth": "1985-03-15",  "documenttype": "russian\_passport",  "documentseries": "4510",  "documentnumber": "123456",  "documentexpirationdate": "2030-03-15",  "fare1": {  " farepriceid ": "3e4f5d6a-7c89-4b0d-9e2a-1a2b3c4d5e6f",  "totalprice": 15500,  "currency": {  "id": "b32f6142-8df9-4a30-980b-45ffb391d5b9",  "code": "usd",  "name": "доллар сша"  }  }  } | bookingclass:        required:          - id          - code        properties:          id:            type: string            format: uuid            example: 02d837dc-9d8d-4c64-955a-8f0e16df9276          name:            type: string            example: эконом          code:            type: string            example: qcl        type: object  passenger:        required:          - passengertype        properties:          id:            type: string            format: uuid            example: 02d837dc-9d8d-4c64-955a-8f0e16df9276          passengertype:            type: string            enum: [adult, child, infant]            example: adult          firstname:            type: string            example: гослинг          lastname:            type: string            example: гослинг          middlename:            type: string            example: иванович          gender:            type: string            enum: [male, female]            example: male          dateofbirth:            type: string            format: date            example: '1980-11-12'          documenttype:            type: string            enum: [russian\_passport, foreign\_passport]            example: russian\_passport          documentseries:            type: string            example: 77          documentnumber:            type: string            example: 123456          documentexpirationdate:            type: string            format: date            example: '1980-11-12'        type: object  contact:        required:          - contacttype        properties:          id:            type: string            format: uuid            example: 02d837dc-9d8d-4c64-955a-8f0e16df9276          contacttype:            type: string            enum: [phone, email]            example: phone          value:            type: string            example: +79999999999        type: object  selectedfareprice:        required:          - totalprice        properties:          id:            type: string            format: uuid            example: 02d837dc-9d8d-4c64-955a-8f0e16df9276          totalprice:            type: number            format: digits            example: 8000.00          currency:            $ref: '#/components/schemas/currency'        type: object   passengerfareprice:        required:          - passengertype        properties:          id:            type: string            format: uuid            example: 02d837dc-9d8d-4c64-955a-8f0e16df9276          passengertype:            type: string            enum: [adult, child, infant]            example: adult          firstname:            type: string            example: райан          lastname:            type: string            example: гослинг          middlename:            type: string            example: иванович          gender:            type: string            enum: [male, female]            example: male          dateofbirth:            type: string            format: date            example: '1980-11-12'          documenttype:            type: string            enum: [russian\_passport, foreign\_passport]            example: russian\_passport          documentseries:            type: string            example: 77          documentnumber:            type: string            example: 123456          documentexpirationdate:            type: string            format: date            example: '1980-11-12'          fare:            $ref: '#/components/schemas/selectedfareprice'        type: object |

# 3)Фильтры и параметры и возможные ответы для каждого endpoint

**POST /flight/list**

**Запрос для получения** массива объектов Flights по фильтрам пользователя. Нам требуются фильтры:

* город или аэропорт отправления;
* город или аэропорт прибытия;
* дата вылета;
* количество пассажиров и их тип;
* критерии для фильтрации;
* критерии для сортировки.

Для удобства мы выделим критерии поиска в отдельное dto - FlightsSearch.

В ответе нам необходимо получить **массив объектов Flights**:

|  |  |
| --- | --- |
| DTO JSON | DTO YAML |
| "flights": [  {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "flightNumber": "0524",  "carrier": {  "id": "e13259d5-7c73-4fb2-8b86-38408d30202c",  "name": "Аэрофлот",  "codeIATA": "SU"  },  "aircraftModel": "Airbus A321",  "departureDate": "2024-10-10",  "departureTime": "10:15:00",  "arrivalDate": "2024-10-10",  "arrivalTime": "11:45:00",  "flightDuration": "1:30",  "departureAirport": {  "id": "acd11a35-9768-4228-9ff6-e66929e4dd8a",  "name": "Шереметьево",  "codeIATA": "SVO"  "countryName": "Россия",  "countryCode": "RU",  "cityName": "Москва",  "cityCode": "MOW",  },  "arrivalAirport": {  "id": "6811f72c-5dab-4156-91d8-ff1ac8a575c2",  "name": "Пулково",  "codeIATA": "LED"  "countryName": "Россия",  "countryCode": " RU",  "cityName": "Санкт-Петербург",  "cityCode": "LED",  },  "fares": [  {  "id": "ada59145-c4bd-4cc8-8c6c-3d09b0d5153f",  "bookingClass": {  "id": "8a5a56d1-807b-48a7-a899-b2d56b6be730",  "code": "QCL",  "name": "Эконом"  }  "name": "С багажом",  "handLuggageWeight": 10,  "handLuggageSize": "55x40x25",  "handLuggageQuantity": 1,  "buggageWeight": 23,  "buggageSize": "203",  "buggageQuantity": 1,  "prices": [  {  "id": "8e60c4d6-0a38-4359-8e97-99ca6fe88bb1",  "passengerType": "ADULT",  "price": "8000",  "currency": {  "id": "29b51335-2473-4739-a6f5-d29eed6fadd2",  "currencyCode": "RUB",  "currencyName": "Российский рубль"  },  {  "id": "8e60c4d6-0a38-4359-8e97-99ca6fe88bb1",  "passengerType": "CHILD",  "price": "6000",  "currency": {  "id": "29b51335-2473-4739-a6f5-d29eed6fadd2",  "currencyCode": "RUB",  "currencyName": "Российский рубль"  },  {  "id": "8e60c4d6-0a38-4359-8e97-99ca6fe88bb1",  "passengerType": "INFANT",  "price": "6000",  "currency": {  "id": "29b51335-2473-4739-a6f5-d29eed6fadd2",  "currencyCode": "RUB",  "currencyName": "Российский рубль"  }  }  ]  }  }  ]  }  ] | /flight/list:  post:  tags:  - flight  summary: searches available flights by user's filters  operationId: searchFlights  description: searches available flights by user's filters  responses:  '200':  description: Search results matching criteria  content:  application/json:  schema:  type: array  items:  $ref: '#/components/schemas/Flight'  '400':  description: bad input parameter  requestBody:  content:  application/json:  schema:  $ref: '#/components/schemas/FlightsSearch'  description: List of flights with prices |

**GET /flight/{flightId}**

|  |  |
| --- | --- |
| DTO JSON | DTO YAML |
| {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "flightNumber": "0524",  "carrier": {  "id": "e13259d5-7c73-4fb2-8b86-38408d30202c",  "name": "Аэрофлот",  "codeIATA": "SU"  },  "aircraftModel": "Airbus A321",  "departureDate": "2024-10-10",  "departureTime": "10:15:00",  "arrivalDate": "2024-10-10",  "arrivalTime": "11:45:00",  "flightDuration": "1:30",  "departureAirport": {  "id": "acd11a35-9768-4228-9ff6-e66929e4dd8a",  "name": "Шереметьево",  "codeIATA": "SVO"  "countryName": "Россия",  "countryCode": "RU",  "cityName": "Москва",  "cityCode": "MOW",  },  "arrivalAirport": {  "id": "6811f72c-5dab-4156-91d8-ff1ac8a575c2",  "name": "Пулково",  "codeIATA": "LED"  "countryName": "Россия",  "countryCode": " RU",  "cityName": "Санкт-Петербург",  "cityCode": "LED",  },  "fares": [  {  "id": "ada59145-c4bd-4cc8-8c6c-3d09b0d5153f",  "bookingClass": {  "id": "8a5a56d1-807b-48a7-a899-b2d56b6be730",  "code": "QCL",  "name": "Эконом"  }  "name": "С багажом",  "handLuggageWeight": 10,  "handLuggageSize": "55x40x25",  "handLuggageQuantity": 1,  "buggageWeight": 23,  "buggageSize": "203",  "buggageQuantity": 1,  "prices": [  {  "id": "8e60c4d6-0a38-4359-8e97-99ca6fe88bb1",  "passengerType": "ADULT",  "price": "8000.00",  "currency": {  "id": "29b51335-2473-4739-a6f5-d29eed6fadd2",  "currencyCode": "RUB",  "currencyName": "Российский рубль"  }  }  ]  }  }  ]  } | /flight/{flightId}:  get:  tags:  - flight  summary: get detailed info about flight  operationId: flightInfo  description: get detailed info about flight  parameters:  - in: path  name: flightId  description: flight ID  required: true  schema:  type: string  format: uuid  responses:  '200':  description: Search results matching criteria  content:  application/json:  schema:  type: $ref: '#/components/schemas/Flight'  '400':  description: bad input parameter |

**POST /booking/create**

Создание нового бронирования. На вход нам необходимо передать объект Booking, который содержит данные по бронированию и связанным сущностям:

* пассажиры;
* рейсы в бронировании;
* выбранные тарифы и их цены;
* контакты для отправки уведомлений.

Таким образом, тело запроса может выглядеть так:

{

"totalPrice": "8000.00",

"contacts": [

{

"contactType": "PHONE",

"value": "+79999999999"

},

{

"contactType": "EMAIL",

"value": "test@test.ru"

}

],

"flights": [

{

"id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",

"flightNumber": "0524",

"quantity": 1,

"totalPrice": "8000.00"

"passengers": [

{

"passengerType": "ADULT",

"firstName": "Райан",

"lastName": "Гослинг",

"middleName": "Иванович",

"gender": "MALE",

"dateOfBirth": "1980-11-12",

"documentType": "RUSSIAN\_PASSPORT",

"documentSeries": "77",

"documentNumber": "123456",

"documentExpirationDate": "2026-01-01",

"fare": {

"farePriceId": "02d837dc-9d8d-4c64-955a-8f0e16df9276",

"totalPrice": "8000.00"

}

}

]

}

]

}

Чтобы убедиться, что все данные были созданы корректно, нам потребуется получить в ответе данные по созданному бронированию и связанным сущностям:

|  |  |
| --- | --- |
| JSON | YAML |
| {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "totalPrice": "8000.00",  "state": "ACTIVE",  "createdAt": "2024-10-10T11:03:15.838Z",  "userId": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "contacts": [  {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "contactType": "PHONE",  "value": "+79999999999"  },  {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "contactType": "EMAIL",  "value": "test@test.ru"  }  ],  "flights": [  {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "flightNumber": "0524",  "quantity": 1,  "totalPrice": "8000.00"  "passengers": [  {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "passengerType": "ADULT",  "firstName": "Райан",  "lastName": "Гослинг",  "middleName": "Иванович",  "gender": "MALE",  "dateOfBirth": "1980-11-12",  "documentType": "RUSSIAN\_PASSPORT",  "documentSeries": "77",  "documentNumber": "123456",  "documentExpirationDate": "2026-01-01",  "fare": {  "farePriceId": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "totalPrice": "8000.00"  }  }  ]  }  ]  } | /booking/create:  post:  tags:  - booking  summary: creates new booking  operationId: createBooking  description: creates new booking  responses:  '200':  description: booking created succesfully  content:  application/json:  schema:  type: array  items:  $ref: '#/components/schemas/Booking'  '400':  description: bad input parameter  requestBody:  content:  application/json:  schema:  $ref: '#/components/schemas/Booking'  description: created booking info |

**GET /booking/list/{userId}**

Получение бронирований пользователя по его id. Здесь мы снова используем GET, потому что поиск бронирований будет осуществляться только по одному параметру:

{userId}

В ответе нам требуется получить массив объектов Booking c userId = uuid пользователю, отправившему запрос:

|  |  |
| --- | --- |
| [  {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "totalPrice": "8000.00",  "state": "ACTIVE",  "createdAt": "2024-10-10T11:03:15.838Z",  "userId": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "contacts": [  {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "contactType": "PHONE",  "value": "+79999999999"  },  {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "contactType": "EMAIL",  "value": "test@test.ru"  }  ],  "flights": [  {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "flightNumber": "0524",  "quantity": 1,  "totalPrice": "8000.00"  "passengers": [  {  "id": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "passengerType": "ADULT",  "firstName": "Райан",  "lastName": "Гослинг",  "middleName": "Иванович",  "gender": "MALE",  "dateOfBirth": "1980-11-12",  "documentType": "RUSSIAN\_PASSPORT",  "documentSeries": "77",  "documentNumber": "123456",  "documentExpirationDate": "2026-01-01",  "fare": {  "farePriceId": "02d837dc-9d8d-4c64-955a-8f0e16df9276",  "totalPrice": "8000.00"  }  }  ]  }  ]  }  ] | Преобразуем в YAML:  /booking/list/{userId}:  get:  tags:  - booking  summary: get list of user's bookings  operationId: bookingList  description: get list of user's bookings  parameters:  - in: path  name: userId  description: user ID  required: true  schema:  type: string  format: uuid  responses:  '200':  description: Search results matching criteria  content:  application/json:  schema:  type: array  items:  $ref: '#/components/schemas/Booking'  '400':  description: bad input parameter |

# 4)Далее была собрана спецификация в swaggerhub