

# Case Study Raporu

Projede standart MVC yapısını kurdum. DB işlerini Repository'ye, bussines logic Service'e, http isteklerini Controller'a bıraktım. Veri tabanı işlemleri için Spring Data JPA kullandım, böylece uzun uzun SQL yazmak yerine findByName gibi methodları kullandım.

Güvenlik ve düzen için dışarıdan gelen verileri direkt Entity sınıflarına almak yerine DTO pattern'ini kullandım. Özellikle bilet alma, buyTicket, kısmında hem bilet oluşturup hem de koltuk sayısını azalttığım için, işlem yarım kalırsa veriler bozulmasın diye Transactional annotation ekledim, bu sayede bir hata olursa yapılan her şey geri alınıyor.

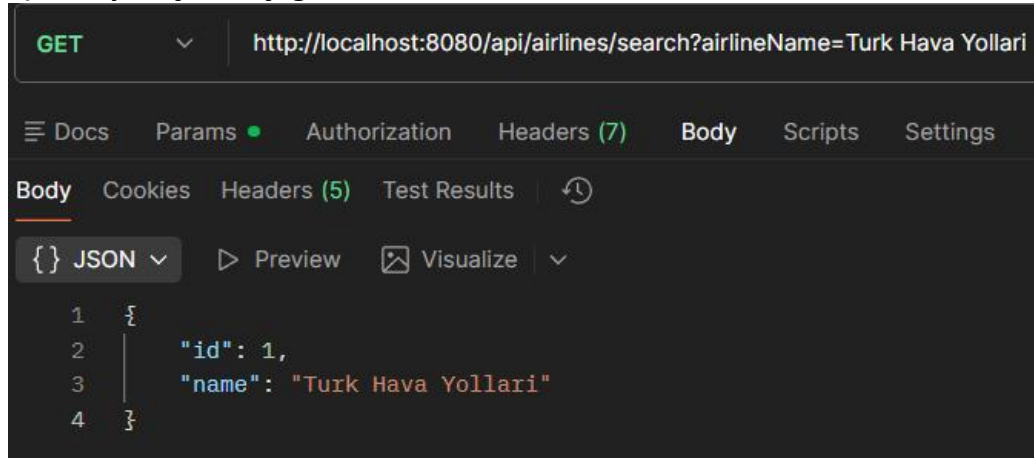
Test tarafında ise işi ikiye ayırdım. Controller testleri için Web MVC Test kullandım çünkü sunucuyu komple kaldırmadan sadece o endpoint'in çalışıp çalışmadığını (200 dönme) görmek istedim. Servis katmanında ise Mockito ile Repository'leri taklit ettim (mocklamak). Gerçek veritabanına gitmeden, sadece yazdığım fiyat hesaplama ve maskeleme mantığının doğru çalışıp çalışmadığını test edebildim.

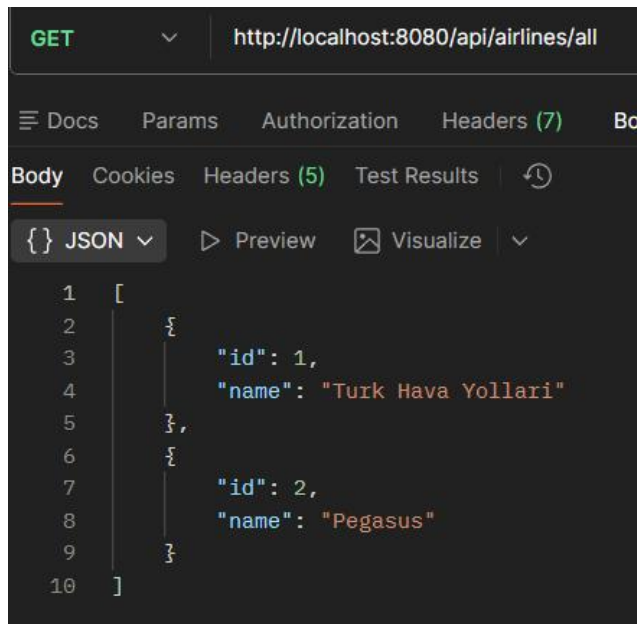
İstenen tüm özellikleri kapsayacak şekilde backend geliştirmelerini tamamladım. Kodun üzerinden geçmek veya yaklaşımıyı detaylandırmak için müsaitliğiniz olduğunda görüşme de yapabiliriz. Sorularınız olursa mail yoluyla da iletişime geçebilirsiniz.

## Postman İstekleri

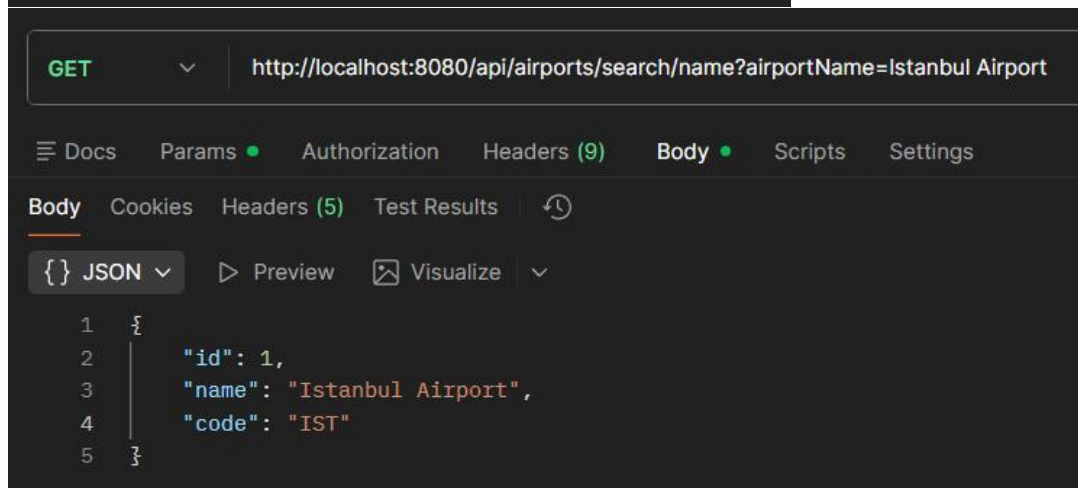
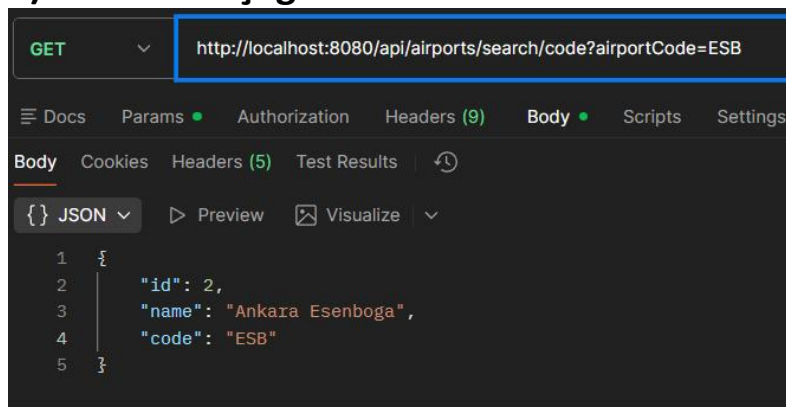
### GET METHODLARI:

#### 1) Havayolu şirketi çağırma:





## 2) Havaalanı çağırma



```
GET http://localhost:8080/api/airports/all

Body
JSON
[
  {
    "id": 1,
    "name": "Istanbul Airport",
    "code": "IST"
  },
  {
    "id": 2,
    "name": "Ankara Esenboga",
    "code": "ESB"
  }
]
```

### 3) Rota Çağırma

```
GET http://localhost:8080/api/routes/search/name?dep=Istanbul Airport&arr=Ankara Esenboga

Body
JSON
[
  {
    "id": 1,
    "departureAirport": {
      "id": 1,
      "name": "Istanbul Airport",
      "code": "IST"
    },
    "arrivalAirport": {
      "id": 2,
      "name": "Ankara Esenboga",
      "code": "ESB"
    }
  }
]
```

```
GET http://localhost:8080/api/routes/search/code?dep=IST&arr=ESB

Body
JSON
[
  {
    "id": 1,
    "departureAirport": {
      "id": 1,
      "name": "Istanbul Airport",
      "code": "IST"
    },
    "arrivalAirport": {
      "id": 2,
      "name": "Ankara Esenboga",
      "code": "ESB"
    }
  }
]
```

#### 4) Uçuş Çağırma

Aynı uçuşu farklı şekillerde çağırabiliriz

```
GET http://localhost:8080/api/flights/search/departure?airportName=Istanbul Airport

GET http://localhost:8080/api/flights/search/arrival?airportName=Ankara Esenboga
```

GET

http://localhost:8080/api/flights/search/airline-name?airlineName=Turk Hava Yollari

Docs

Params

Authorization

Headers (9)

Body

Scripts

Settings

Body

Cookies

Headers (5)

Test Results



{ } JSON

Preview

Visualize



```
1  [
2    {
3      "id": 1,
4      "airline": {
5        "id": 1,
6        "name": "Turk Hava Yollari"
7      },
8      "route": {
9        "id": 1,
10       "departureAirport": {
11         "id": 1,
12         "name": "Istanbul Airport",
13         "code": "IST"
14       },
15       "arrivalAirport": {
16         "id": 2,
17         "name": "Ankara Esenboga",
18         "code": "ESB"
19       }
20     },
21     "basePrice": 1000.0,
22     "seat": 10,
23     "soldSeats": 3
24   }
25 ]
```

## 5) Bilet Çağırma

GET <http://localhost:8080/api/tickets/52000acb-d1e3-42fc-b7a2-9ef805e7233b>

Docs Params Authorization Headers (7) **Body** Scripts Settings

☒ none ☐ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

This request does not have a body.

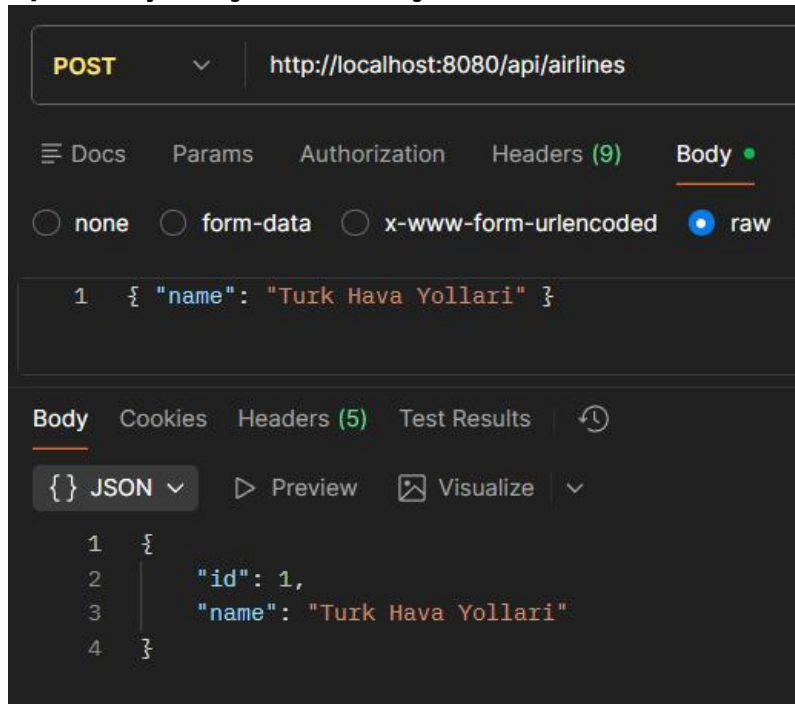
Body Cookies Headers (5) Test Results

**{ } JSON** Preview Visualize

```
8      "name": "Turk Hava Yollari"
9    },
10   "route": {
11     "id": 1,
12     "departureAirport": {
13       "id": 2,
14       "name": "Istanbul Airport",
15       "code": "IST"
16     },
17     "arrivalAirport": {
18       "id": 1,
19       "name": "Ankara Esenboga",
20       "code": "ESB"
21     }
22   },
23   "basePrice": 1000.0,
24   "seat": 10,
25   "soldSeats": 3
26 },
27 "creditCardMasked": "123456*****3456",
28 "pricePaid": 1200.0,
29 "canceled": false
30 }
```

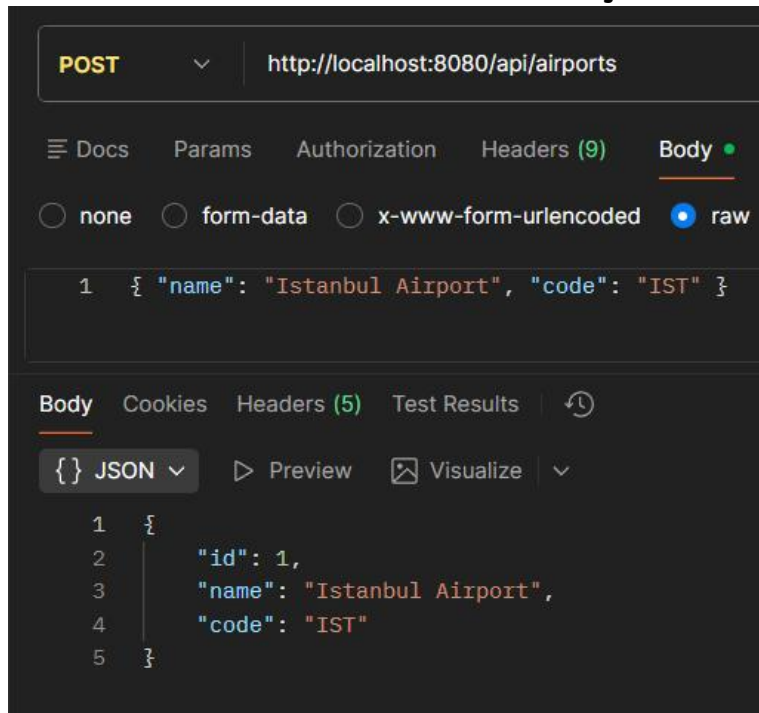
# POST METHODLARI:

## 1) Havayolu şirketi Oluşturma:



## 2) Havaalanı Oluşturma:

IST ve ESB olmak üzere 2 tane oluşturdum



### 3) Rota Oluşturma:

The screenshot displays a REST client interface with the following details:

- Method:** POST
- URL:** http://localhost:8080/api/routes
- Body Type:** raw (selected)
- Request Body (JSON):**

```
1 {
2   "departureAirportCode": "IST",
3   "arrivalAirportCode": "ESB"
4 }
```
- Response Body (JSON):**

```
1 {
2   "id": 1,
3   "departureAirport": {
4     "id": 1,
5     "name": "Istanbul Airport",
6     "code": "IST"
7   },
8   "arrivalAirport": {
9     "id": 2,
10    "name": "Ankara Esenboga",
11    "code": "ESB"
12  }
13 }
```

#### 4) Uçuş Tanımlama:

The screenshot displays a REST client interface with a POST request to `http://localhost:8080/api/flights`. The request body is a JSON object with the following fields:

```
{
  "airlineName": "Turk Hava Yollari",
  "routeId": 1,
  "basePrice": 1000.0,
  "seat": 10
}
```

The response is shown in JSON format, containing the following fields:

```
{
  "route": {
    "id": 1,
    "departureAirport": {
      "id": 2,
      "name": "Istanbul Airport",
      "code": "IST"
    },
    "arrivalAirport": {
      "id": 1,
      "name": "Ankara Esenboga",
      "code": "ESB"
    }
  },
  "basePrice": 1000.0,
  "seat": 10,
  "soldSeats": 0
}
```

## 5) Bilet Satın Alma:

```
POST http://localhost:8080/api/tickets/buy?flightId=1&creditCard=1234-5678 9012-3456

Body
JSON
{
  "ticketNumber": "bc1636d8-0574-4706-b6db-69b3eec4750d",
  "flight": {
    "id": 2,
    "airline": {
      "id": 1,
      "name": "Turk Hava Yollari"
    },
    "route": {
      "id": 1,
      "departureAirport": {
        "id": 2,
        "name": "Istanbul Airport",
        "code": "IST"
      },
      "arrivalAirport": {
        "id": 1,
        "name": "Ankara Esenboga",
        "code": "ESB"
      }
    },
    "basePrice": 1000.0,
    "seat": 10,
    "soldSeats": 1
  },
  "creditCardMasked": "123456*****3456",
  "pricePaid": 1000.0,
  "canceled": false
}
```

Bileti Tekrar satın alırken fiyat artışı:

```
    },
    "creditCardMasked": "123456*****3456",
    "pricePaid": 1100.0,
    "canceled": false
  },
```

```
    "creditCardMasked": "123456*****3456",
    "pricePaid": 1200.0,
    "canceled": false
  },
}
```

## Ticket Delete Methodu:

```
DELETE http://localhost:8080/api/tickets/52000acb-d1e3-42fc-b7a2-9ef805e7233b

25     "soldSeats": 2
26   },
27   "creditCardMasked": "123456*****3456",
28   "pricePaid": 1200.0,
29   "canceled": true
30 }
```

## H2 DB:

jdbc:h2:mem:testdb

AIRLINE

AIRPORT

FLIGHT

ROUTE

TICKET

INFORMATION\_SCHEMA

Users

H2 2.3.232 (2024-08-11)

RunRun SelectedAuto complete

SELECT \*  
FROM AIRLINE

SELECT \*  
FROM AIRLINE;  

ID	NAME
1	Turk Hava Yollari

(1 row, 2 ms)

Edit

jdbc:h2:mem:testdb

AIRLINE

AIRPORT

FLIGHT

ROUTE

TICKET

INFORMATION\_SCHEMA

Users

H2 2.3.232 (2024-08-11)

RunRun SelectedAuto complete

SELECT \*  
FROM AIRPORT

SELECT \*  
FROM AIRPORT;  

ID	CODE	NAME
1	IST	Istanbul Airport
2	ESB	Ankara Esenboga

(2 rows, 1 ms)

jdbc:h2:mem:testdb

AIRLINE

AIRPORT

FLIGHT

ROUTE

TICKET

INFORMATION\_SCHEMA

Users

H2 2.3.232 (2024-08-11)

RunRun SelectedAuto completeClear

SQL statement:

SELECT \*  
FROM FLIGHT

SELECT \*  
FROM FLIGHT;  

ID	BASE_PRICE	SEAT	SOLD_SEATS	AIRLINE_ID	ROUTE_ID
1	1000.0	10	3	1	1

(1 row, 2 ms)

Edit

jdbc:h2:mem:testdb

AIRLINE

AIRPORT

FLIGHT

ROUTE

TICKET

INFORMATION\_SCHEMA

Users

H2 2.3.232 (2024-08-11)

RunRun SelectedAuto completeClear

SQL statement:

SELECT \*  
FROM ROUTE

SELECT \*  
FROM ROUTE;  

ID	ARRIVAL_AIRPORT_ID	DEPARTURE_AIRPORT_ID
1	2	1

(1 row, 0 ms)

Edit

jdbc:h2:mem:testdb

AIRLINE

AIRPORT

FLIGHT

ROUTE

TICKET

INFORMATION\_SCHEMA

Users

H2 2.3.232 (2024-08-11)

RunRun SelectedAuto completeClear

SQL statement:

SELECT \*  
FROM TICKET

SELECT \*  
FROM TICKET;  

ID	CREDIT_CARD_MASKED	IS_CANCELED	PRICE_PAID	TICKET_NUMBER	FLIGHT_ID
1	123456*****3456	FALSE	1000.0	7dafa832-128d-42f6-9bd8-dfd76b26e401	1
2	123456*****3456	FALSE	1100.0	84743294-676c-44c4-83d8-b2686c03540a	1
3	123456*****3456	FALSE	1200.0	5559edb7-993e-42f3-91b5-dc2737f02557	1

(3 rows, 1 ms)

Edit