# Full Stack Development Case Study - Java Developer 1

## Introduction

Aim of the project is to implement a system for aviation industry. This system should be able to calculate all the possible routes from point A to point B to provide users better experience while booking their flights.

The requirements are the followings:

- A database for storing data should be initialized. Any database (PostgreSQL, MySQL, MSSQL or H2) is allowed for use.
- A Spring Boot Java REST API should be implemented.
- Hibernate framework should be used as an ORM framework to map Java classes and database tables.
- Swagger support for API documentation should be provided, and Swagger UI should be reachable.

## Database Requirements

In context of the project, the following entities should be defined in database:

### Locations

Represents the locations which will be displayed in “from” and “to” dropdown boxes while searching flights. Locations entities should contain the following fields:

- **Name**
- **Country**
- **City**
- **Location code**
    - For airports, location codes are the 3 characters long IATA codes. Example: **SAW** for Sabiha Gökçen Airport, **IST** for istanbul airport. (For searching IATA code of a particular airport: https://www.iata.org/en/publications/directories/code-search/)
    - For other locations, any coding pattern can be used (example: CCIST for İstanbul City Center)

### Transportations

Represents the transportations from one location to another. Transportation entities should contain the following fields:

- **Origin Location**
- **Destination Location**
- **Transportation Type:** FLIGHT, BUS, SUBWAY, UBER

# Full Stack Development Case Study - Java Developer 1

## API Requirements

A Spring Boot REST API should be implemented and combined with Hibernate for database integration. Through REST controllers, API needs to provide endpoints for:

- CRUD operations for locations
- CRUD operations for transportations
- Returning all the valid **route**s from one location to another.

### Definition of Route

- A route can be defined as a sequence of connected transportations. Each route can have 3 connected transportations:
  - **Before flight transfer**: Optional. Transportation type must be other than FLIGHT
  - **Flight**: Mandatory. Transportation type must be FLIGHT.
  - **After flight transfer**: Optional. Transportation type must be other than FLIGHT.
- To consider two transportations "connected", the arrival location of the prior transportation should match the departure location of the subsequent transportation.

  Examples:
  - A bus ride from **Taksim Square** to **İstanbul Airport** and a flight from **İstanbul Airport** to **London Heathrow Airport** are connected transportations. ✓
  - Funicular from **Kabataş Pier** to **Taksim Square** and a flight from **İstanbul Airport** to **London Heathrow Airport** are not connected transportations. ✗

### Limitations

A sequence of connected transportations cannot be considered as a valid route if:

- There are more than 3 transportations from origin to destination.
- There is no flight among them
- There are more than one flights among them
- There are more than one before flight transfers among them
- There are more than one after flight transfers among them

Valid route examples:

- UBER ➞ FLIGHT ➞ BUS ✓
- FLIGHT ➞ BUS ✓
- UBER ➞ FLIGHT ✓
- FLIGHT ✓

Invalid route examples:

- UBER ➞ BUS ➞ FLIGHT ✕ (multiple before flight transfers)
- UBER ➞ BUS ✕ (no flight)
- UBER ➞ FLIGHT ➞ FLIGHT ✕ (multiple flights)
- FLIGHT ➞ FLIGHT ✕ (multiple flights)
- FLIGHT ➞ SUBWAY➞ UBER ✕ (multiple after flight transfers)

An example valid route “Taksim square ➞ Wembley Stadium” would look like this:

- A **bus ride** from **Taksim Square** to **İstanbul Airport**
- A **flight** from **İstanbul Airport** to **London Heathrow Airport**
- A **Uber ride** from **London Heathrow Airport** to **Wembley Stadium**

IMPORTANT: If there are multiple transfer options for the same route, all should be returned as if they are separate routes. Example:

- Taksim Square **(UBER)** ➞ Istanbul Airport ➞ London Heathrow Airport **(BUS)** ➞ Webley Stadium
- Taksim Square **(UBER)** ➞ Istanbul Airport ➞ London Heathrow Airport **(UBER)** ➞ Webley Stadium
- Taksim Square **(SUBWAY)** ➞ Istanbul Airport ➞ London Heathrow Airport **(BUS)** ➞ Webley Stadium
- Taksim Square **(SUBWAY)** ➞ Istanbul Airport ➞ London Heathrow Airport **(UBER)** ➞ Webley Stadium
- Taksim Square **(BUS)** ➞ Istanbul Sabiha Gökçen Airport ➞ London Heathrow Airport **(BUS)** ➞ Webley Stadium
- Taksim Square **(BUS)** ➞ Istanbul Sabiha Gökçen Airport ➞ London Heathrow Airport **(UBER)** ➞ Webley Stadium

As it would be noticed, the first 4 routes are the same routes since all the stops in the middle (Istanbul and Heathrow Airports) are same. But they are listed as 4 different routes because there are 2 different transfer options for before flight and after flight transfers each.

## Additional Complexity (Bonus)

This addition is one of the Nice to Have’s. An extra field “Operating Days” can be added to “transportation” entities. This field should be an array of integers representing on which days of the week the corresponding transportation is operated. The value of the array being [1, 3, 5, 6] means that the corresponding transportation is only active on Monday, Wednesday, Friday and Saturday.

To make “Operating Days” field meaningful, an extra “date” parameter (apart from “origin” and “destination”) should also be taken in the endpoint which returns the all the valid routes. If a route has any transportation which is not operated in the “date”, then it should be excluded from the valid routes list.

**Example:**

A route has 3 transportations:

- Bus ride from Taksim Square to Istanbul Airport which has operating days value **[2, 3, 5]**
- Flight from Istanbul Airport to London Heathrow Airport which has operating days value **[2, 3, 7]**
- Uber ride from London Heathrow Airport to Webley Stadium which has operating days value **[3, 5]**

If the paremeter “date” is selected as:

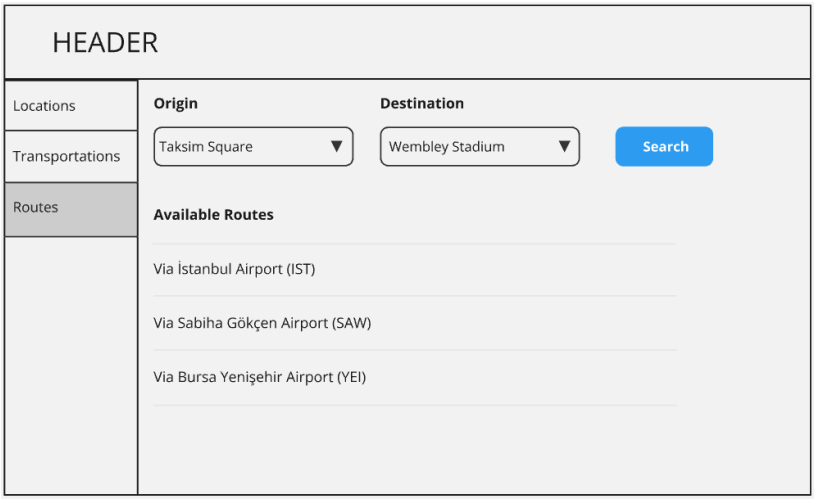
- “12 March 2025”, then the route mentioned above should be returned in the result list. ✓
- “11 March 2025” instead, then the route should not be returned in the result list. Since 11 March is Tuesday which is not among the operating days of the after flight transfer. ✗

## Front-end Requirements

A responsive SPA with header and side bar menu should be implemented in React.

There should be 3 links in the side bar

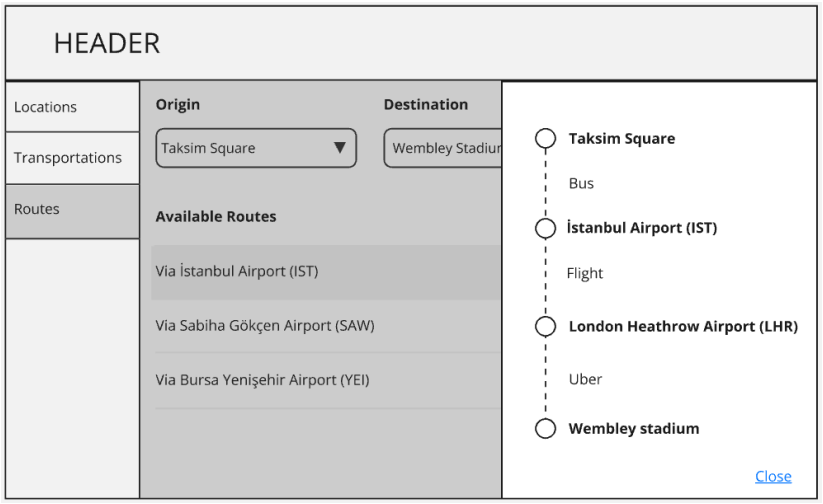
- “Locations” for listing, creating, updating and deleting locations.
- “Transportations” for listing, creating, updating and deleting transportations.
- “Routes” for listing all valid routes between two different locations.
- In “Routes” page, origin and destination locations are selected on dropdown boxes. And the available routes should be displayed in a listing view. (The following design is just a mock for suggestion, there is no restriction).




**Important:** If it is decided to implement the Additional Complexity, then an extra datepicker input should be added to the design provided above.

# Full Stack Development Case Study - Java Developer 1

- After clicking an available route, route details should be displayed in a side panel (alternatives will also be accepted).




## "Nice-to-Have"s

- Back-end validations on api.
- A Postman collection including sample requests.
- Spring Boot tests or JUnit tests for testing the endpoint which returns all valid routes.
- Implementing the Additional Complexity