Ticket to Ride Web Application
Product Requirements

for “Java for Beginners” course

v.0.0.1

You are chosen to create a web application called "**Ticket to ride**".

Main objects:

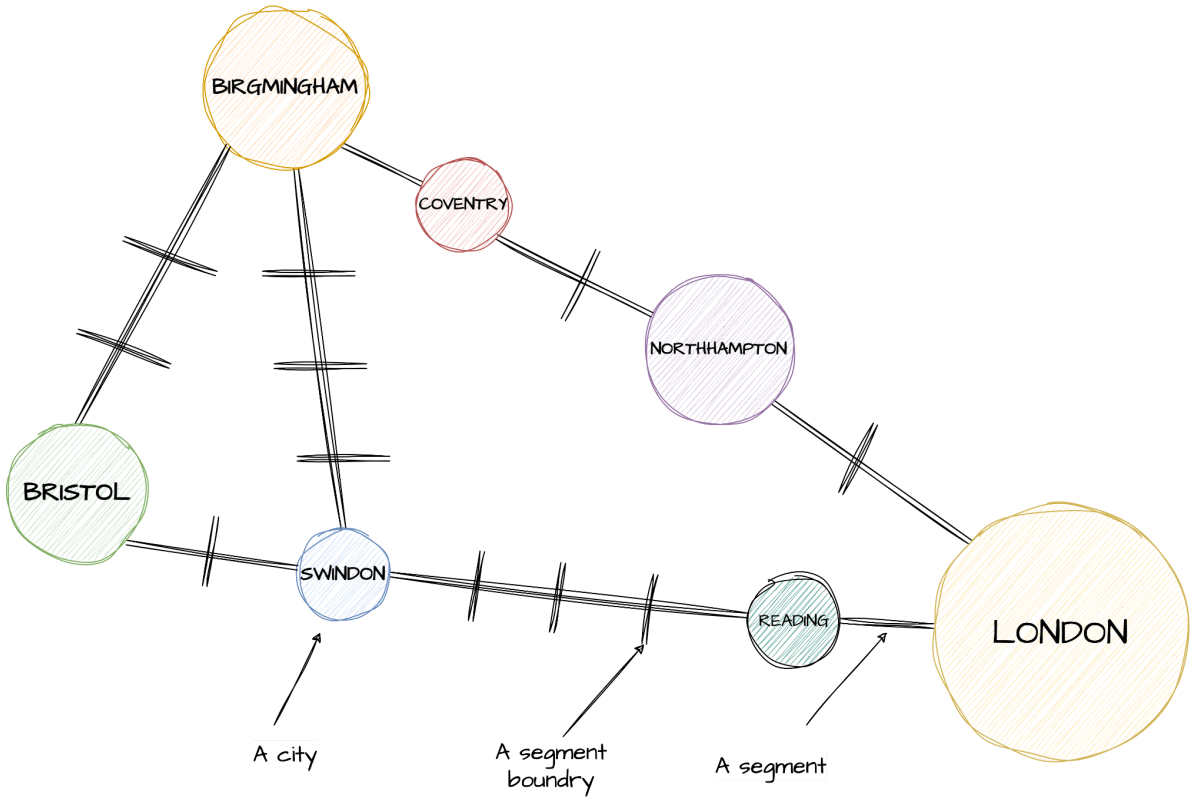
Segment - part of a route.

Route - set of segments (1+), representing a route between two towns.

Ticket - a permission for a Traveller to travel via a certain route. Has a price.

Traveller - a person who takes a ticket from town A to town B.

The price of the travel through 1 segment is 5 GBP.
The price of the travel through 2 segments is 7 GBP.
The price of the travel through 3 segments is 10 GBP.



You need to provide an API with 2 functions (endpoints) for a traveller:

1. Calculate the price of a most optimal travel between two towns in GBP.
2. Save the ticket to a storage if a traveller has enough money.

Requirements:

- The API to calculate the price should be public.
- The service should have layered architecture.
- The service should be able to save successfully bought ticket.

Nice to have:

- An ability to persist the ticket to a database.
- The API to buy a ticket should be private. It should be protected with username and password.
- Logging.
- Java Docs
- Readme file

Stack:

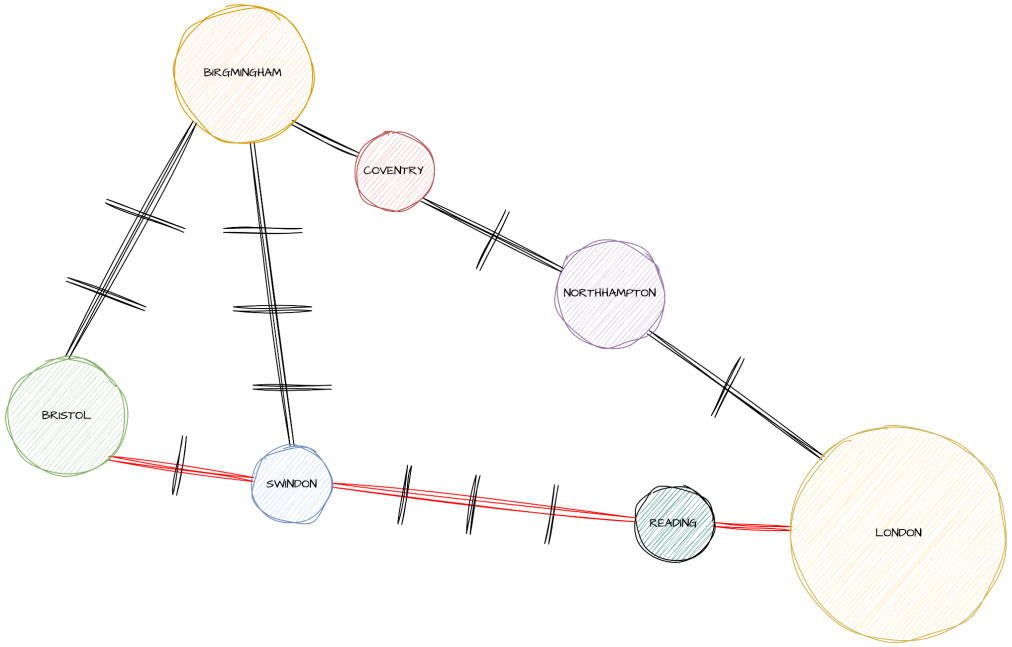
Java 21, Spring Boot, Spring Data, Spring Security, JUnit 5, Mockito, PostgreSQL, Maven/Gradle

Examples

Find a ticket:

Example input 1:
{"departure":"London","arrival":"Bristol"}
Example output 1:
{"segments":7,"price":25,"currency":"GBP"}

Description: 3x2 segments **discount** at price of 10x2 and 1 segment at price of 5.



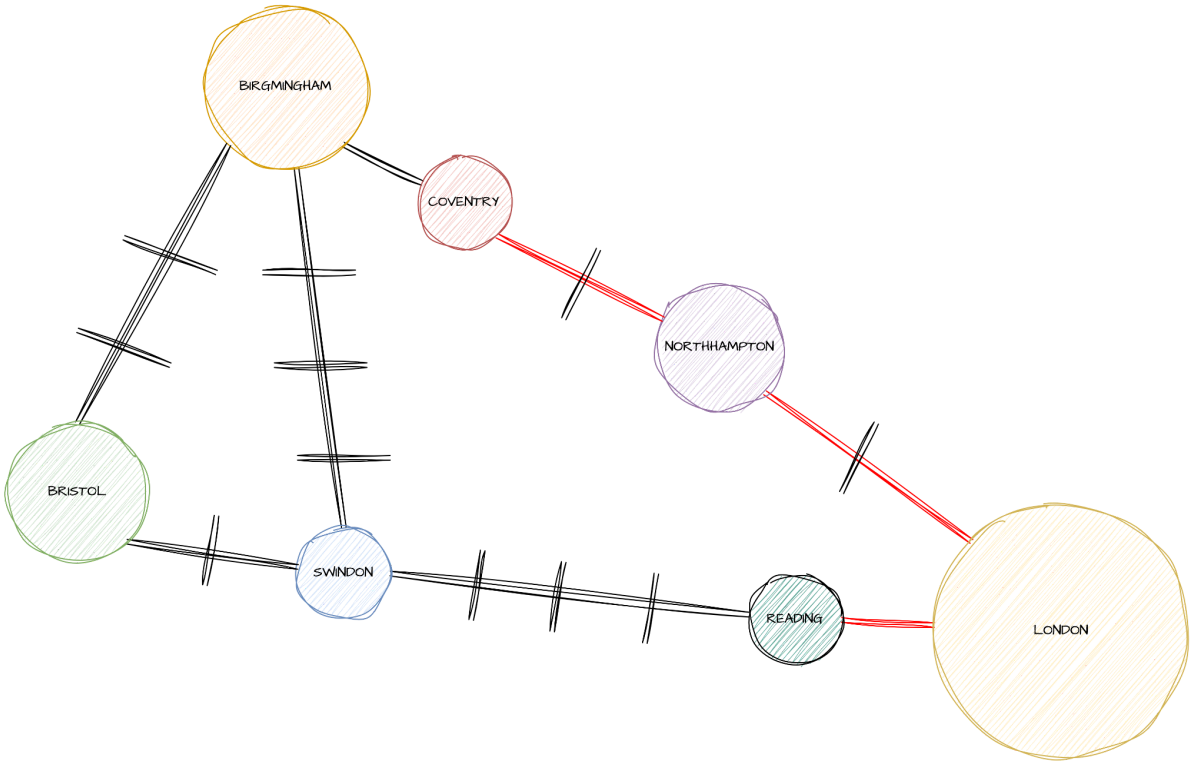
Save a ticket:

Example input 1:
{"departure":"London","arrival":"Bristol", "segments":7,"price":25,"currency":"GBP",
"travellerAmount":30,"traveller":"John Doe"}
Example output 1:
{"result":"success","change":5,"currency":"GBP"}

Find a ticket:

Example input 2:
{"departure":"Coventry","arrival":"Reading"}
Example output 2:
{"segments":5,"price":17,"currency":"GBP"}

Description: 3 segments **discount** at price of 10 and 2 segments **discount** at price of 7.



Save a ticket:

Example input 1:
{"departure":"Coventry","arrival":"Reading", "segments":5,"price":17,"currency":"GBP",
"travellerAmount":15,"traveller":"John Doe"}
Example output 1:
{"result":"failure","lackOf":2,"currency":"GBP"}