Create a runnable Java application which implements a RESTful service

The service must expose the following functionality (see page 2 for data models)

1. Create a new wallet
    a. This endpoint must create a new wallet and store it in the database, returning an ID
2. Retrieve a wallet
3. Make a cash deposit to a given wallet
    a. This endpoint must add the cash amount to the wallet's current balance
4. Retrieve a fare for a given origin and destination train station
    a. Dynamically retrieve fare pricing from http://api.bart.gov (details below)
5. Buy a ticket with a given wallet for a given origin and destination train station
    a. This endpoint must save the ticket to the given wallet and subtract the fare from the wallet's current balance

Requirements

1. Use Spring Boot as the application framework
2. Your application must dynamically retrieve fare pricing from http://api.bart.gov (details below)
3. Store the data (wallet) in a database of your choice (H2 is acceptable)
4. Provide instructions on how to run your application, a docker-compose file is a plus
5. Unit and/or Integration tests are a plus

BART API Details

1. Your application must use http://api.bart.gov/docs/sched/fare.aspx for fare pricing
    a. Only worry about fares between one station and another, you don't have to worry about scheduling or anything else
    b. Here is the list of station name abbreviations you can use for the API http://api.bart.gov/docs/overview/abbrev.aspx
    c. For the API key, they have provided a free key in their documentation so that you don't have to sign up for anything: MW9S-E7SL-26DU-VV8V
    d. Here is an example api call for 12th St. Oakland City Center to Embarcadero: http://api.bart.gov/api/sched.aspx?cmd=fare&orig=12th&dest=embr&date=today&key=MW9S-E7SL-26DU-VV8V&json=y

Suggestions

You will be evaluated on your API design as well as overall project structure and completeness. If you feel you do not have enough time to complete the assignment (try not to spend more than 3 hours on this), give priority to having a running functional application. Feel free to be creative with your solution and API endpoints and we will discuss afterwards any pitfalls you encountered along the way and how you would go about changing your code if you had more time.

Data Model Suggestions

1. Create a new wallet (Response body)
    a. This should return an ID, UUID, or a URL to the new wallet.
       For example: /wallets/6c5dc7e0-7946-461d-aa56-14a860aa76c7

2. Retrieve a wallet (Response body)

```
{
    "balance": 15.00
    "tickets": [
        {
            "origin": "12th",
            "destination": "embr"
        },
        {
            "origin": "dubl",
            "destination": "dbrk"
        }
    ]
}
```

3. Make a cash deposit to a given wallet (Request body)

```
{
    "amount": 25.00
}
```

4. Retrieve a fare for a given origin and destination train station (Response body)

```
{
    "origin": "12th",
    "destination": "embr",
    "fare": 3.70
}
```

5. Buy a ticket for a given wallet, origin, and destination train station (Response body)

```
{
    "origin": "12th",
    "destination": "embr"
}
```