Mercedes_Event_Driven_Fuel_Cost_Calculator

Tools and environments used :-

1.Spring/Spring Boot Framework.
2.Maven build tool.
3. RabbitMq - message-broker software.
4.Tomcat server(to test locally)
5.Mysql Database.
6.spring tool suite.

API details:-

1.Fetch fuel price API
Created Rest api for any random city fuel price. Used mock values which are stored in Database.
Endpoint url :- http://localhost:8080/fuelPrice/check
Sample Request:- {"CITY":"Bangalore"}
Sample Response:- {"PRICE": "96","STATUS": "SUCCESS"}

2.Fetch event API
Created Rest api for fuel lid event, where the event can be triggered manually.
Endpoint url:- http://localhost:8081/fuelEvent/check
Sample Request:- {"fuellid":true,"city":"Bangalore"}
Sample Response:-
```
{
 "fuel_quantity": "4.0",
 "total_amount": "384.0",
  "external_msg": "",
  "response_code": "Success"}
```

Event driven service:-

Used **RabbitMq - message-broker software** for handling the events.

Code flow:-

1.For every two minutes event sender will send a fuel lid status with city name. Depending upon the fuel lid status, fuel price api will get called and it returns the fuel price of that city. Assuming that the car can take 1 liter/30 seconds , total amount and fuel quantity for 2 minutes will be calculated.
Used Threads for invoking sender for every 2 minutes.
2. Event can be triggered manually by Fetch event api . used @cacheable for cache purpose.
3.Used mock data for City names and fuel prices,Which may get selected randomly.
4. Price and fuel quantity are logged inside the event receiving service.
5.Fuel API project contains the exposed API which will return the fuel cost based on City
6.Event Project contains the Event Sender and Event Receiver functionalities

ToDo:-

1.API request and response should be encrypted for secure access . encryption and decryption methods are attached with source code.

Asumption:-

1.Car can take 1 liter/30 seconds.
2.Total price and quantity of fuel calculated for 2 minutes.