Walkthrough on Sweet-home Project

Introduction:

- Developed 3 microservices namely Eureka-server, Booking and Payment
- Used h2 in-memory database with below properties for configuration in Payment and Booking service
- ```
  spring.datasource.url=jdbc:h2:~/test;DB_CLOSE_ON_EXIT=FALSE;AUTO_SERVER
  =TRUE
  spring.datasource.driverClassName=org.h2.Driver
  spring.datasource.username=sa
  spring.datasource.password=
  spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
  spring.h2.console.enabled=true
  spring.jpa.hibernate.ddl-auto = update
  ```
- As API-Gateway was optional, ignored that microservice
- Added proper comments where required for better understanding

Eureka-Server:

- Added dependency "spring-cloud-starter-netflix-eureka-server" and annotation @EnableEurekaServer
- Added below configuration to ignore eureka server adding itself to the registry.
  eureka.client.register-with-eureka=false
  eureka.client.fetch-registry=false
- Application runs on 8761 port.
- Url: http://localhost:8761/

Booking Service:

- Added dependency "spring-cloud-starter-netflix-eureka-client" and annotation @EnableDiscoveryClient for eureka-server to find the client
- Also added web, h2, data-jpa dependencies
- Application runs on port 8081 port
- Url: http://localhost:8081/
- H2 Url: http://localhost:8081/h2-console
- Added exception scenarios.
- Microservice contains 2 endpoints described as below:
  1. /hotel/booking (POST)
     - This endpoint saves all booking related data to booking table. It generates the roomPrice based on the numberOfRooms and toDate/fromDate. Also contains logic/method to generate roomNumbers separated by "," based of numberOfRooms field.
  ```

2. /hotel/booking/{bookingId}/transaction (POST)
    - This endpoint calls payment service "/payment/transaction" endpoint. Used DiscoveryClient bean to get the payment service baseurl from eureka registry without hard coding the same. Created RestTemplate bean to make synchronous API call to payment service. This method calls payment service to generate random transactionId. Also for provided path variable bookingId saves the transaction corresponding to that bookingId. Returns booking entity object as response.

Payment Service:

- Added dependency "spring-cloud-starter-netflix-eureka-client" and annotation @EnableDiscoveryClient for eureka-server to find the client
- Also added web, h2, data-jpa dependencies
- Application runs on port 8083 port
- Url: http://localhost:8083/
- H2 Url: http://localhost:8081/h2-console
- Microservice contains 2 endpoints described as below:
    1. /payment/transaction (POST)
        - This endpoint saves the transaction data/entity and returns the transactionId for the saved object.
    2. /payment/transaction/{transactionId} (GET)
        - This endpoint fetches transaction entity for the given transactionId.