**Spring Boot Hotel Booking Project**

This is a Hotel booking project in Java / Maven / Spring Boot application.In-memory H2 data base used for the demonstraton purpose.

**How to Run**

This application is packaged as a jar which has Tomcat 8 embedded. No Tomcat or JBoss installation is necessary. You run it using the java -jar command.

Clone this repository

Make sure you are using JDK 1.8 and Maven 3.x

You can build the project and run the tests by running mvn clean package

Once successfully build, you can run the service by one of these two methods:

java -jar <jar-path>

or

mvn spring-boot:run

Check the stdout to make sure no exceptions are thrown

Once the application runs you should see something like this

2020-09-26 21:24:57.695 INFO 25192 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 9090 (http) with context path ''

2020-09-26 21:24:58.025 INFO 25192 --- [ restartedMain] c.m3bi.booking.HotelBookingApplication : Started HotelBookingApplication in 7.229 seconds (JVM running for 7.739)

Once Application Start successfully open swagger documentation of resources

Swagger URL: http://localhost:9090/swagger-ui/index.html

**How to Test**

Clone this repository

Go to the root folder of the clone directory.

Execute “mvn clean test”.

Test Case are added for :

Create Booking

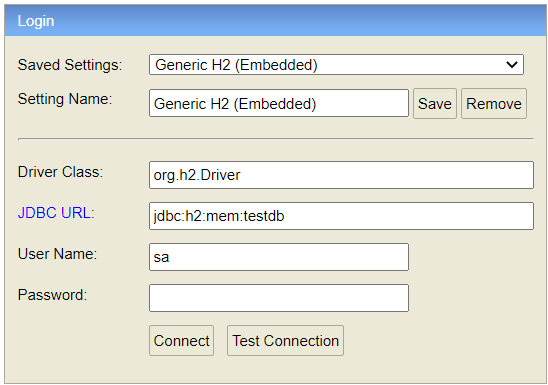
Check Booking Status

Update Bonus

**About In-Memory database**

In Data base access url: http://localhost:9090/h2-console

JDBC URL: jdbc:h2:mem:testdb



**Database Tables:**

**User**

|  |  |  |
| --- | --- | --- |
| Id | Name | BonusId |
| Integer | String | Integer |

**Hotel**

|  |  |  |
| --- | --- | --- |
| Id | Status | Price |
| Integer | String | Integer |

**Bonus**

|  |  |
| --- | --- |
| Id | Points |
| Integer | Integer |

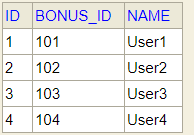
**Pending\_Approval**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Hotel\_Id | Price | Status | User\_Id |
| Integer | Integer | Integer | String | Integer |

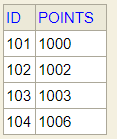
**Booking**

|  |  |  |
| --- | --- | --- |
| Id | Hotel\_Id | User\_Id |
| Integer | Integer | Integer |

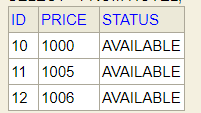
User Table



Bonus Table



Hotel Table



Pending\_Approval Table

Store booking entries with status pending for approval due to low bonus points.

Booking Table

Store Booking entries of final booking entries.

Note: Currently for Demonstration fixed set of data is used. Please refer data.sql file in src/main/resources

Mapping between User and Bonus table is not present. Any addition and deletion of records need to be manage manually.

**About the Service**

The service is just a simple hotel booking REST service. It uses an in-memory database (H2) to store the data.

If your database connection properties work, you can call some REST endpoints defined in com.m3bi.booking.controller on port 9090.

All APIs are documented by Swagger3 using annotations.

**Hotel Booking Resource**

**Book hotel**

POST /bookings/user/{userid}/hotel/{hotelid}

**Get Booking Status**

GET /bookings/status/user/{userid}

**Update Bonus**

PUT /bonuses/user/{userid}

**Scenario 1**: User1 with n bonus points, booking hotel with n price and status AVAILABLE.

**RESULT**: Book Hotel and update hotel status as BOOKED. Add entry in Booking Table.

E.g. POST http://localhost:9090/bookings/user/1/hotel/10

Response: Your Hotel Is Booked. Booking ID: 1 user id: 1

**Scenario 2**: User2 with m bonus points, booking hotel with n price (m<n) and status AVAILABLE. **RESULT**: Hold Hotel and update hotel status PENDINGAPPROVAL. Add entry in Pending\_Approval table.

E.g. POST http://localhost:9090/bookings/user/2/hotel/11

Response: Hotel Pending For Approval

**Scenario 3**: User3 with m bonus points, booking hotel with n price (m<n) and status AVAILABLE. **RESULT**: Hold Hotel and update hotel status PENDINGAPPROVAL. Add entry in Pending\_Approval table.

E.g. POST http://localhost:9090/bookings/user/3/hotel/12

Response: Hotel Pending For Approval

**Scenario 4**: User 4 with n bonus points. Booking Hotel with price n and status PENDINAPPROVAL(suppose hotel is already booked by user3).

**RESULT**: Delete previous booking from Pending\_Approval Table. Book Hotel and update hotel status as BOOKED.

E.g. POST http://localhost:9090/bookings/user/4/hotel/12

Response: Your Hotel Is Booked. Booking ID: 2 user id: 4

**Scenario 5**: Update Bonus of User2.

**RESULT**: Update Bonus of User2, check booking status of user if booking status id PENDINGAPPROVAL and bonus is greater than price of hotel. Change booking status from PENDINGAPPROVAL to BOOKED. Delete entry from Pending\_Approval Table and add entry in Booking Table.

E.g. check user booking status

GET http://localhost:9090/bookings/status/user/2

Response: Your booking request is pending for approval

Update Bonus

PUT http://localhost:9090/bonuses/user/2

RequestBody: Contains user bonusid and update bonus ponts

{

"id": 102,

"points": 1100

}

Response: In Response header update resource link <http://localhost:9090/bonuses/102>

Check booking status again

GET <http://localhost:9090/bookings/status/user/2>

Response: Your Hotel Is Booked. Booking ID: 3 user id: 2