**Spring Boot REST CRUD Application with POSTMAN Client**

**POSTMAN Tool**

Postman is an interactive and automatic tool for verifying the APIs of the project. Postman is a Google Chrome App for interacting with HTTP APIs

**Download POSTMAN**

url: <https://www.postman.com/downloads>

**HTTP Methods**

GET – Read Operation (Select Command)

POST – Write / Create Operation (Insert Command)

PUT – Update Operation (Update Command)

DELETE – Delete Operation (Delete Command)

* Create table “ticket” in MySQL

mysql>create table ticket (ticket\_id int(3),passenger\_name varchar(10), source\_station varchar(15), dest\_station varchar(15), email varchar(20));

* Create a Spring Starter Project in STS

Name : SpringBootRESTCRUDProj

Java Version: 8

Group : com.rest.springboot

Artifact: RestSpringBootCRUDProj

Package : com.rest.springboot

Click Next

Add the following project dependencies

* + Spring Web
  + Spring Data JPA
  + MySQL Driver
  + Lombok

Click Finish

* Create a package “com.rest.springboot.entities” in src/main/java folder
* Create a Java Bean class “Ticket” in “com.rest.springboot.entities” package

Ticket.java

**package** com.rest.springboot.entities;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.Id;

**import** javax.persistence.Table;

**import** lombok.AllArgsConstructor;

**import** lombok.Data;

**import** lombok.NoArgsConstructor;

@Entity

@Table(name="ticket")

@Data

@AllArgsConstructor

@NoArgsConstructor

**public** **class** Ticket {

@Id

@Column(name="ticket\_id")

**private** Integer ticketId;

@Column(name="passenger\_name",nullable=**false**)

**private** String passengerName;

@Column(name="source\_station")

**private** String sourceStation;

@Column(name="dest\_station")

**private** String destStation;

@Column(name="email")

**private** String email;

}

* Create a package “com.rest.springboot.dao” in src/main/java folder
* Create an interface “TicketBookingDAO” in com.rest.springboot.dao package

TicketBookingDAO.java

package com.rest.springboot.dao;

import org.springframework.data.jpa.repository.JpaRepository;

import com.rest.springboot.entities.Ticket;

public interface TicketBookingDAO extends JpaRepository<Ticket, Integer>{

}

* Create a package “com.rest.springboot.service” in src/main/java folder
* Create a Service class “TicketBookingService” in com.rest.springboot.service package

TicketBookingService.java

**package** com.rest.springboot.service;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Service;

**import** com.rest.springboot.dao.TicketBookingDAO;

**import** com.rest.springboot.entities.Ticket;

@Service

**public** **class** TicketBookingService {

@Autowired

**private** TicketBookingDAO ticketBookingDAO;

**public** Ticket createTicket(Ticket ticket) {

**return** ticketBookingDAO.save(ticket);

}

**public** Ticket getTicketById(Integer ticketId) {

**return** ticketBookingDAO.findById(ticketId).get();

}

**public** Iterable<Ticket> getAllBookedTickets() {

**return** ticketBookingDAO.findAll();

}

**public** **void** deleteTicket(Integer ticketId) {

ticketBookingDAO.deleteById(ticketId);

}

**public** Ticket updateTicket(Integer ticketId, String newEmail) {

Ticket ticketFromDb = ticketBookingDAO.findById(ticketId).get();

ticketFromDb.setEmail(newEmail);

Ticket upadedTicket = ticketBookingDAO.save(ticketFromDb);

**return** upadedTicket;

}

}

* Create a package “com.rest.springboot.controller” in src/main/java folder
* Create a Rest Controller class “TicketBookingController” in com.rest.springboot.controller package

TicketBookingController.java

**package** com.rest.springboot.controller;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.web.bind.annotation.DeleteMapping;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.PathVariable;

**import** org.springframework.web.bind.annotation.PostMapping;

**import** org.springframework.web.bind.annotation.PutMapping;

**import** org.springframework.web.bind.annotation.RequestBody;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.RestController;

**import** com.rest.springboot.entities.Ticket;

**import** com.rest.springboot.service.TicketBookingService;

@RestController

@RequestMapping(value="/api/tickets")

**public** **class** TicketBookingController {

@Autowired

**private** TicketBookingService ticketBookingService;

@PostMapping(value="/create")

**public** Ticket createTicket(@RequestBody Ticket ticket){

**return** ticketBookingService.createTicket(ticket);

}

@GetMapping(value="/ticket/{ticketId}")

**public** Ticket getTicketById(@PathVariable("ticketId")Integer ticketId){

**return** ticketBookingService.getTicketById(ticketId);

}

@GetMapping(value="/ticket/alltickets")

**public** Iterable<Ticket> getAllBookedTickets(){

**return** ticketBookingService.getAllBookedTickets();

}

@DeleteMapping(value="/ticket/{ticketId}")

**public** **void** deleteTicket(@PathVariable("ticketId")Integer ticketId){

ticketBookingService.deleteTicket(ticketId);

}

@PutMapping(value="/ticket/{ticketId}/{newEmail:.+}")

**public** Ticket updateTicket(@PathVariable("ticketId")Integer ticketId,@PathVariable("newEmail")String newEmail){

**return** ticketBookingService.updateTicket(ticketId,newEmail);

}

}

* Update application.properties file of src/main/resources folder

application.properties

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.datasource.url=jdbc:mysql://localhost:3306/mydb

spring.datasource.username=root

spring.datasource.password=root

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQLDialect

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

* Run the project

Right click on RestSpringBookCRUDProj -> Run As -> Spring Boot App

Test the application using POSTMAN

Open POSTMAN and do the following requests

Create Request

POST <http://localhost:8080/api/tickets/create>

In Body

Select JSON

{“ticketId”:”1”, “passengerName”:”Raj”,”sourceStation”:”Hyderabad”,

“destStation”:”Delhi”,”email”:”raj@gmail.com”}

Click Send

Add some more records

{“ticketId”:”2”, “passengerName”:”Ramu”,”sourceStation”:”Chennai”,

“destStation”:”Mumbai”,”email”:”ramu@gmail.com”}

{“ticketId”:”3”, “passengerName”:”Ramya”,”sourceStation”:”Kochi”,

“destStation”:”Hyderabad”,”email”:”ramya@gmail.com”}

GET by Id Request

GET <http://localhost:8080/api/tickets/ticket/1>

GET <http://localhost:8080/api/tickets/ticket/2>

GET All Tickets Request

GET <http://localhost:8080/api/tickets/ticket/alltickets>

DELETE by Id Request

DELETE <http://localhost:8080/api/tickets/ticket/1>

UPDATE Request

PUT <http://localhost:8080/api/tickets/ticket/2/ramu_new@gmail.com>