# Java Date Time – How to build SpringBoot RestApi – Post/Get request with Java Date Time using Jackson and Make Query with Spring JPA example

In the tutorial, we build a SpringBoot RestAPIs example that post/get data with `java.util.Date` time and save it to MySQL/PostgreSQL database using Spring JPA. Working with Java Date Time is an exciting part but also not easy task, fortunately we have the supporting from utilities of Jackson lib, now the job can be done in an easy way.

Let’s do details by steps!

## Format Java Date Time with Jackson

### Set the Format with @JsonFormat

With the `@JsonFormat` annotation of Jackson, we can use it to format a specific field in Java model:

```
public class DateTimeModel {

    @JsonFormat(pattern="yyyy-MM-dd")
    private Date date;

    @JsonFormat(pattern="yyyy-MM-dd HH:mm:ss")
    private Date datetime;
```

```
        ...
}
```

### Set TimeZone with @JsonFormat

For setting Time Zone, we use `timezone` attribute of the `@JsonFormat`:

```
@JsonFormat(pattern="yyyy-MM-dd HH:mm:ss", timezone="Europe/Paris")
private Date datetimewithzone;
```

### Set Default Format

We can configure a default format & time-zone for all dates in `application.properties`:

```
spring.jackson.date-format=MM-dd-yyyy HH:mm:ss
spring.jackson.time-zone=Asia/Ho_Chi_Minh
```

## Map Date and Time with Spring JPA

We use the `@Temporal` annotation to specify what the field represents. This annotation goes with a parameter having a value of `TemporalType` enum:

```
public enum TemporalType {

    /** Map as java.sql.Date */
    DATE,

    /** Map as java.sql.Time */
    TIME,
    /** Map as java.sql.Timestamp */
    TIMESTAMP
}
```

-> Now, our model will be like:

```
@Entity
@Table(name = "datetimemodel")
public class DateTimeModel {
        @Id
        @GeneratedValue(strategy = GenerationType.AUTO)
        private long id;

    @Column
    @Temporal(TemporalType.DATE)
    @JsonFormat(pattern="yyyy-MM-dd")
    private Date date;

    @Column
    @Temporal(TemporalType.TIMESTAMP)
    @JsonFormat(pattern="yyyy-MM-dd HH:mm:ss")
    private Date datetime;

    @Column
    @Temporal(TemporalType.TIMESTAMP)
    @JsonFormat(pattern="yyyy-MM-dd HH:mm:ss", timezone="Europe/Paris")
    private Date datetimewithzone;

        ...
}
```

## Query the Entities with Date Time property from Database

For query data with Date Time property from database, we need to implement `CrudRepository` and use 2 approaches.

– Use the built-in rule of Spring JPA mechanics:

```
public interface DateTimeRepository extends CrudRepository{

        List findAllByDatetimeBetween(
                                                        Date dateTimeStart,
                                                        Date dateTimeEnd);

        ...
}
```

– Use `@Query` annotation:

```
public interface DateTimeRepository extends CrudRepository{

        ...

    @Query("select d from DateTimeModel d where d.datetime <= :datetime")
    List findAllWithDatetimeBefore(
                @Param("datetime") Date datetime);
}
```

## RequestParam with Date Time in RestApi

For adding a request param with Date Time data type in RestAPI, we use `@RequestParam` & `@DateTimeFormat`

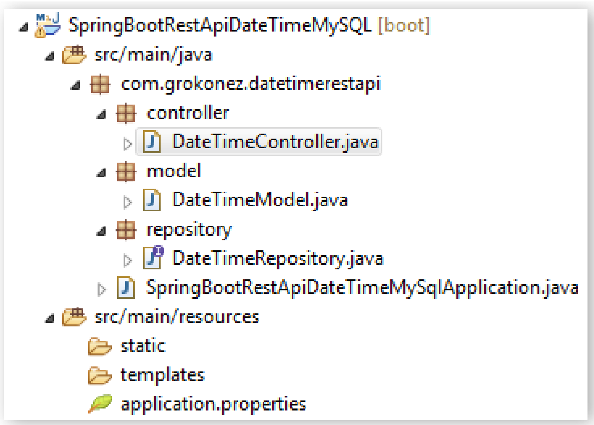
```
@GetMapping("/getallbydatetimebetween")
public List getAllByDatetimeBetween(
                @RequestParam("startdate") @DateTimeFormat(pattern="yyyy-MM-dd HH:mm:ss") Date startdate,
                @RequestParam("enddate") @DateTimeFormat(pattern="yyyy-MM-dd HH:mm:ss") Date enddate) {

        ...
}
```

## Practice

### Create SpringBoot Project

We create a SpringBoot project as below structure:



```
SpringBootRestApiDateTimeMySQL [boot]
  src/main/java
    com.grokonez.datetimerestapi
      controller
        DateTimeController.java
      model
        DateTimeModel.java
      repository
        DateTimeRepository.java
      SpringBootRestApiDateTimeMySqlApplication.java
  src/main/resources
    static
    templates
    application.properties
```

– Dependencies List:

```


        org.springframework.boot
        spring-boot-starter-web


    com.fasterxml.jackson.core
    jackson-core


  org.springframework.boot
  spring-boot-starter-data-jpa


        mysql
        mysql-connector-java
        runtime

```

## Create Dates & Times Model

– Create `DateTimeModel.java` model:

```
package com.grokonez.datetimerestapi.model;

import java.util.Date;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;

import com.fasterxml.jackson.annotation.JsonFormat;

@Entity
@Table(name = "datetimemodel")
public class DateTimeModel {
        @Id
        @GeneratedValue(strategy = GenerationType.AUTO)
        private long id;

    @Column
    @Temporal(TemporalType.DATE)
    @JsonFormat(pattern="yyyy-MM-dd")
    private Date date;

    @Column
    @Temporal(TemporalType.TIMESTAMP)
    @JsonFormat(pattern="yyyy-MM-dd HH:mm:ss")
    private Date datetime;

    @Column
    @Temporal(TemporalType.TIMESTAMP)
    @JsonFormat(pattern="yyyy-MM-dd HH:mm:ss", timezone="Europe/Paris")
    private Date datetimewithzone;

    @Column
    @Temporal(TemporalType.TIMESTAMP)
    private Date defaultformatdatetime;

    public DateTimeModel() {}
    public DateTimeModel(Date date, Date datetime, Date datetimewithzone, Date defaultformatdatetime) {
        this.date = date;
        this.datetime = datetime;
        this.datetimewithzone = datetimewithzone;
        this.defaultformatdatetime = defaultformatdatetime;
    }

    // Getter/Setter date
    public void setDate(Date date) {
        this.date = date;
    }

    public Date getDate() {
        return this.date;
    }

    // Getter/Setter datetime
    public void setDatetime(Date datetime) {
        this.datetime = datetime;
    }

    public Date getDatetime() {
        return this.datetime;
    }

    // Getter/Setter datetimewithzone
    public void setDatetimewithzone(Date datetimewithzone) {
        this.datetimewithzone = datetimewithzone;
    }

    public Date getDatetimewithzone() {
        return this.datetimewithzone;
    }

    // Getter/Setter defaultformatdatetime
    public void setDefaultformatdatetime(Date defaultformatdatetime) {
        this.defaultformatdatetime = defaultformatdatetime;
    }

    public Date getDefaultformatdatetime() {
        return this.defaultformatdatetime;
    }
}
```

### Create Repository

– Create `DateTimeRepository.java` repository:

```
package com.grokonez.datetimerestapi.repository;

import java.util.Date;
import java.util.List;

import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.CrudRepository;
import org.springframework.data.repository.query.Param;

import com.grokonez.datetimerestapi.model.DateTimeModel;

public interface DateTimeRepository extends CrudRepository{

        List findAllByDatetimeBetween(
                                                                Date dateTimeStart,
                                                                Date dateTimeEnd);

    @Query("select d from DateTimeModel d where d.datetime <= :datetime")
    List findAllWithDatetimeBefore(
                @Param("datetime") Date datetime);
}
```

### Create RestAPIs POST/GET/QUERY

– Create `DateTimeController.java` restapis:

```
package com.grokonez.datetimerestapi.controller;
```

```java
import java.util.Date;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.format.annotation.DateTimeFormat;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

import com.grokonez.datetimerestapi.model.DateTimeModel;
import com.grokonez.datetimerestapi.repository.DateTimeRepository;

@RestController
@RequestMapping("/api")
public class DateTimeController {

        @Autowired
        DateTimeRepository dateTimeRepository;

        @GetMapping("/getdatetime")
        public Iterable getDateTimeModel() {
                return dateTimeRepository.findAll();
        }

        @PostMapping("/postdatetime")
        public String postDateTimeMode(@RequestBody DateTimeModel datetime) {
                dateTimeRepository.save(datetime);
                return "Done!";
        }

        @GetMapping("/getallbydatetimebetween")
        public List getAllByDatetimeBetween(
                        @RequestParam("startdate") @DateTimeFormat(pattern="yyyy-MM-dd HH:mm:ss") Date startdate,
                        @RequestParam("enddate") @DateTimeFormat(pattern="yyyy-MM-dd HH:mm:ss") Date enddate) {

                return dateTimeRepository.findAllByDatetimeBetween(startdate, enddate);
        }

        @GetMapping("/getallwithdatetimebefore")
        public List getAllWithDatetimeBefore(
                        @RequestParam("datetime") @DateTimeFormat(pattern="yyyy-MM-dd HH:mm:ss") Date datetime){

                return dateTimeRepository.findAllWithDatetimeBefore(datetime);
        }
}
```

## Database Configuration

### Config MySQL

– `application.properties`:

```
spring.datasource.url=jdbc:mysql://localhost:3306/gkzdb
spring.datasource.username=root
spring.datasource.password=12345
spring.jpa.generate-ddl=true

spring.jackson.date-format=MM-dd-yyyy HH:mm:ss
spring.jackson.time-zone=Asia/Ho_Chi_Minh
```

### Config PostgreSQL

– `application.properties`:

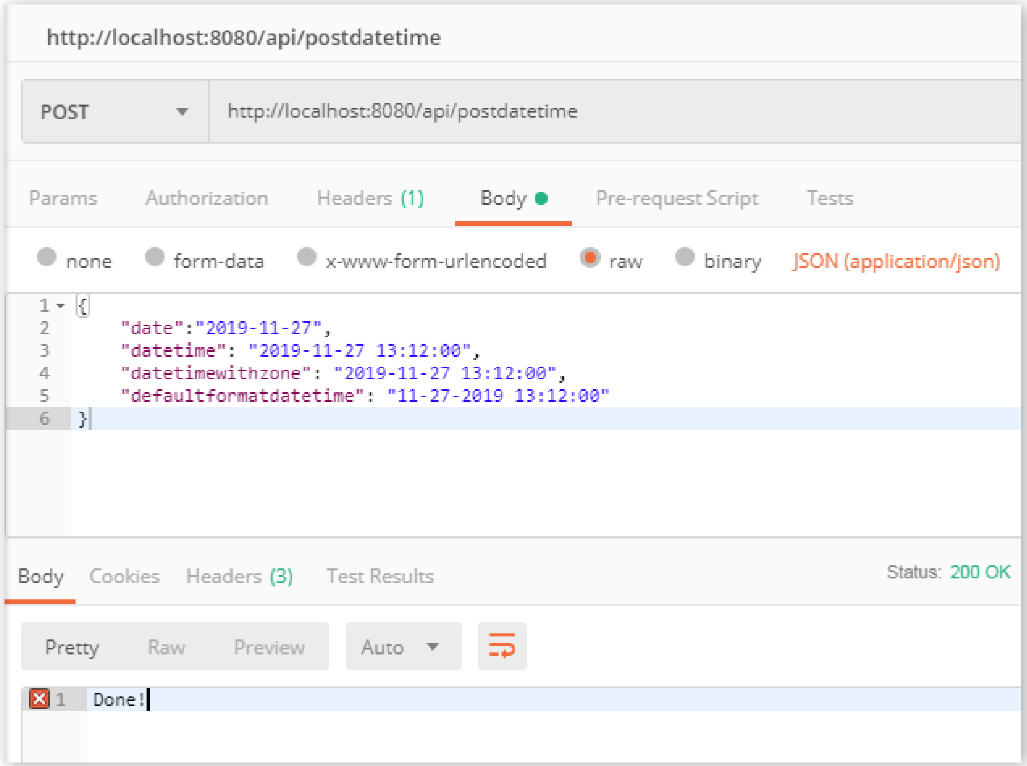
```
spring.datasource.url=jdbc:postgresql://localhost/testdb
spring.datasource.username=postgres
spring.datasource.password=123
spring.jpa.generate-ddl=true

spring.jackson.date-format=MM-dd-yyyy HH:mm:ss
spring.jackson.time-zone=Asia/Ho_Chi_Minh
```

## Run & Check Result

Run SpringBoot project, then makes requests.

– Post request:



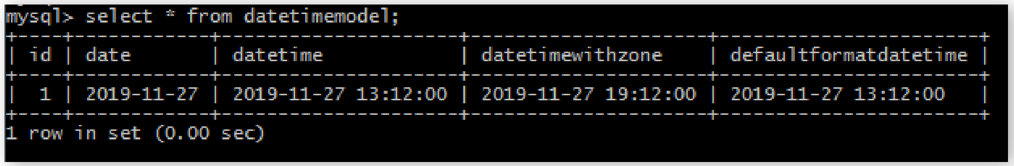
POST http://localhost:8080/api/postdatetime

Body: raw, JSON (application/json)

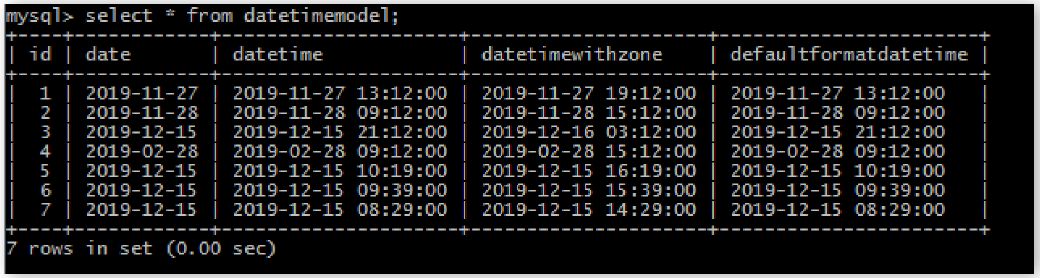
```
{
    "date":"2019-11-27",
    "datetime": "2019-11-27 13:12:00",
    "datetimewithzone": "2019-11-27 13:12:00",
    "defaultformatdatetime": "11-27-2019 13:12:00"
}
```

Status: 200 OK

```
Done!
```

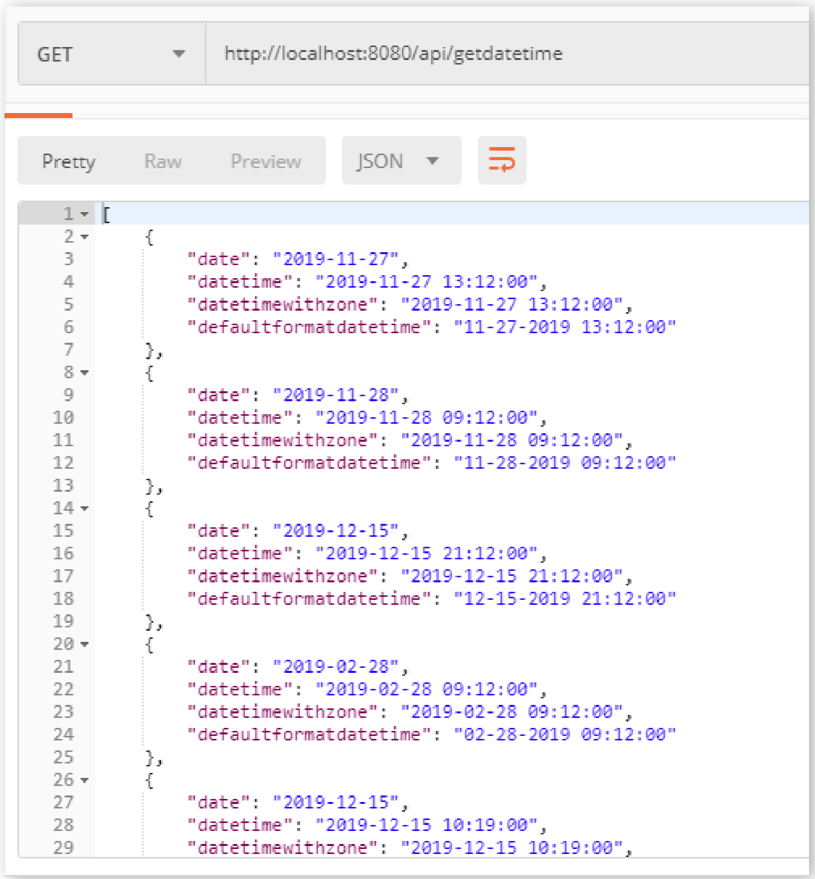


```
mysql> select * from datetimemodel;
+----+------------+---------------------+---------------------+-----------------------+
| id | date       | datetime            | datetimewithzone    | defaultformatdatetime |
+----+------------+---------------------+---------------------+-----------------------+
|  1 | 2019-11-27 | 2019-11-27 13:12:00 | 2019-11-27 19:12:00 | 2019-11-27 13:12:00   |
+----+------------+---------------------+---------------------+-----------------------+
1 row in set (0.00 sec)
```



```
mysql> select * from datetimemodel;
+----+------------+---------------------+---------------------+-----------------------+
| id | date       | datetime            | datetimewithzone    | defaultformatdatetime |
+----+------------+---------------------+---------------------+-----------------------+
|  1 | 2019-11-27 | 2019-11-27 13:12:00 | 2019-11-27 19:12:00 | 2019-11-27 13:12:00   |
|  2 | 2019-11-28 | 2019-11-28 09:12:00 | 2019-11-28 15:12:00 | 2019-11-28 09:12:00   |
|  3 | 2019-12-15 | 2019-12-15 21:12:00 | 2019-12-16 03:12:00 | 2019-12-15 21:12:00   |
|  4 | 2019-02-28 | 2019-02-28 09:12:00 | 2019-02-28 15:12:00 | 2019-02-28 09:12:00   |
|  5 | 2019-12-15 | 2019-12-15 10:19:00 | 2019-12-15 16:19:00 | 2019-12-15 10:19:00   |
|  6 | 2019-12-15 | 2019-12-15 09:39:00 | 2019-12-15 15:39:00 | 2019-12-15 09:39:00   |
|  7 | 2019-12-15 | 2019-12-15 08:29:00 | 2019-12-15 14:29:00 | 2019-12-15 08:29:00   |
+----+------------+---------------------+---------------------+-----------------------+
7 rows in set (0.00 sec)
```

– Get All Entities request:

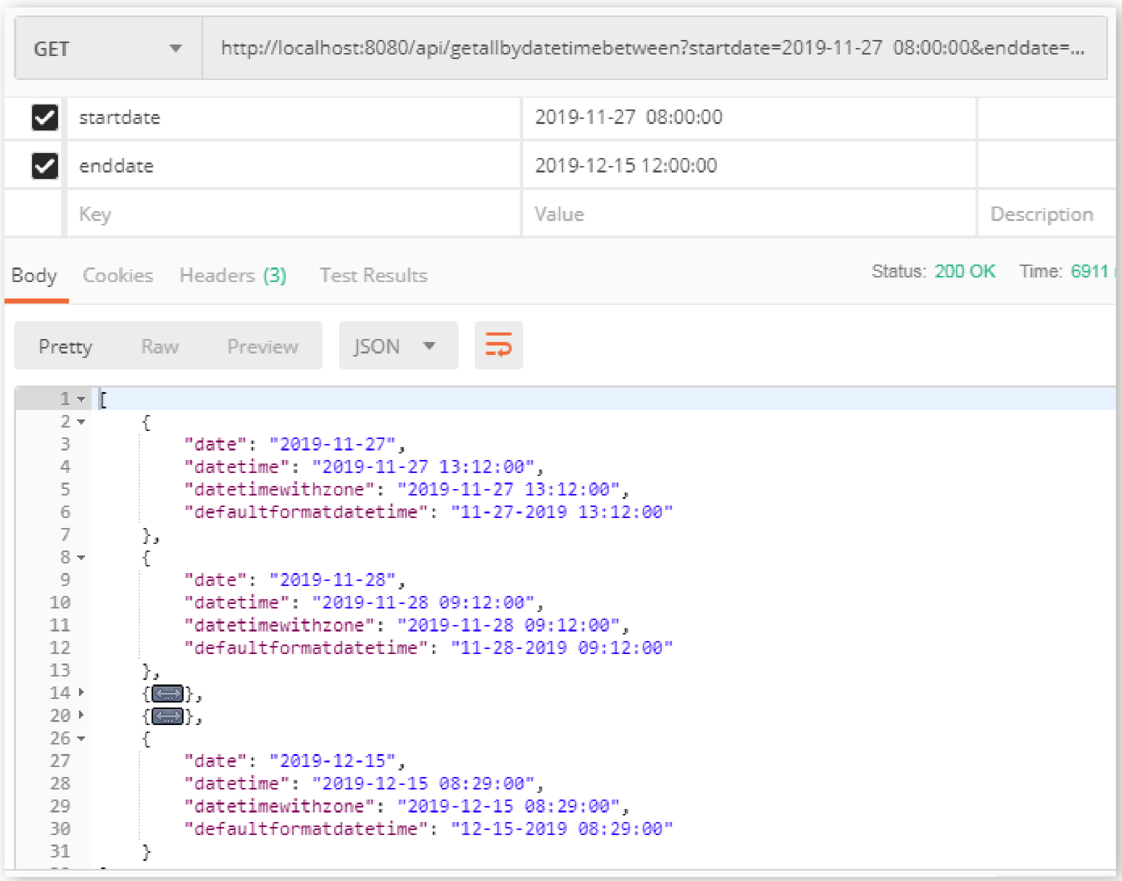


http://localhost:8080/api/getdatetime

```
[
    {
        "date": "2019-11-27",
        "datetime": "2019-11-27 13:12:00",
        "datetimewithzone": "2019-11-27 13:12:00",
        "defaultformatdatetime": "11-27-2019 13:12:00"
    },
    {
        "date": "2019-11-28",
        "datetime": "2019-11-28 09:12:00",
        "datetimewithzone": "2019-11-28 09:12:00",
        "defaultformatdatetime": "11-28-2019 09:12:00"
    },
    {
        "date": "2019-12-15",
        "datetime": "2019-12-15 21:12:00",
        "datetimewithzone": "2019-12-15 21:12:00",
        "defaultformatdatetime": "12-15-2019 21:12:00"
    },
    {
        "date": "2019-02-28",
        "datetime": "2019-02-28 09:12:00",
        "datetimewithzone": "2019-02-28 09:12:00",
        "defaultformatdatetime": "02-28-2019 09:12:00"
    },
    {
        "date": "2019-12-15",
        "datetime": "2019-12-15 10:19:00",
        "datetimewithzone": "2019-12-15 10:19:00",
```

– Query Request ->

+ Get Entities between 2 Date Time:



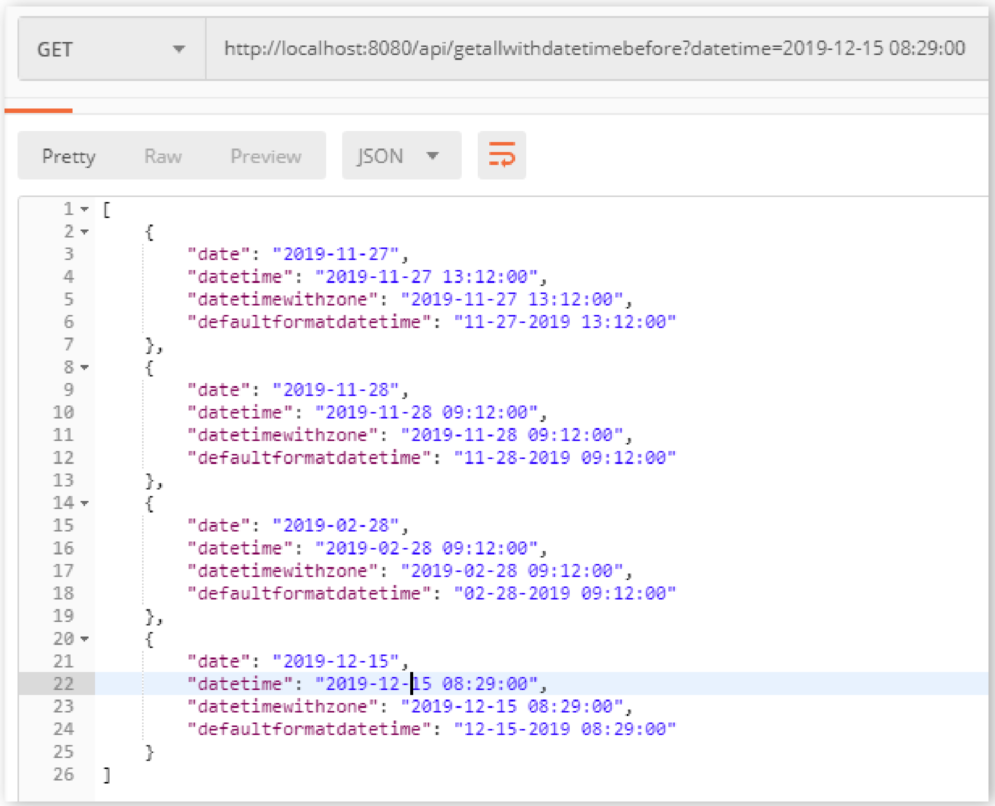
GET http://localhost:8080/api/getallbydatetimebetween?startdate=2019-11-27 08:00:00&enddate=...

| | Key | Value | Description |
|---|---|---|---|
| ✓ | startdate | 2019-11-27 08:00:00 | |
| ✓ | enddate | 2019-12-15 12:00:00 | |

Status: 200 OK Time: 6911

```
[
    {
        "date": "2019-11-27",
        "datetime": "2019-11-27 13:12:00",
        "datetimewithzone": "2019-11-27 13:12:00",
        "defaultformatdatetime": "11-27-2019 13:12:00"
    },
    {
        "date": "2019-11-28",
        "datetime": "2019-11-28 09:12:00",
        "datetimewithzone": "2019-11-28 09:12:00",
        "defaultformatdatetime": "11-28-2019 09:12:00"
    },
    {↔},
    {↔},
    {
        "date": "2019-12-15",
        "datetime": "2019-12-15 08:29:00",
        "datetimewithzone": "2019-12-15 08:29:00",
        "defaultformatdatetime": "12-15-2019 08:29:00"
    }
```

+ Get Entities before a Date Time:



```
http://localhost:8080/api/getallwithdatetimebefore?datetime=2019-12-15 08:29:00
```

```json
[
    {
        "date": "2019-11-27",
        "datetime": "2019-11-27 13:12:00",
        "datetimewithzone": "2019-11-27 13:12:00",
        "defaultformatdatetime": "11-27-2019 13:12:00"
    },
    {
        "date": "2019-11-28",
        "datetime": "2019-11-28 09:12:00",
        "datetimewithzone": "2019-11-28 09:12:00",
        "defaultformatdatetime": "11-28-2019 09:12:00"
    },
    {
        "date": "2019-02-28",
        "datetime": "2019-02-28 09:12:00",
        "datetimewithzone": "2019-02-28 09:12:00",
        "defaultformatdatetime": "02-28-2019 09:12:00"
    },
    {
        "date": "2019-12-15",
        "datetime": "2019-12-15 08:29:00",
        "datetimewithzone": "2019-12-15 08:29:00",
        "defaultformatdatetime": "12-15-2019 08:29:00"
    }
]
```

### Sourcecode

[SpringBootRestApiDateTimeMySQL]

## Conclusion

Through the tutorial, We had learned how to create a SpringBoot RestAPI with Java Date Time data:

- Using `@JsonFormat` annotation to format Date Time
- Using `@Temporal` annotation to map Date and Time to work with Spring JPA
- Query the Entities with Date Time property from Database
- Format RequestParam with Date Time type in RestAPI by using `@DateTimeFormat`

Thank you for reading! See you next time!

By grokonez | March 25, 2019.

## Related Posts

- Spring Boot 2.1 + Angular 8 + MySQL example | Angular HTTP Client + RestAPIs + Spring JPA CRUD + MySQL tutorial
- Vue.js + Spring Boot example | Spring Data JPA + REST + MariaDB CRUD
- Spring Security JWT Authentication example – RestAPIs SpringBoot + Spring MVC + Spring JPA + MySQL
- Spring Boot + Vue.js example | Spring Data JPA + REST + PostgreSQL CRUD
- Spring JPA/Hibernate One-to-Many Association + PostgreSQL | SpringBoot CRUD RestAPIs Post/Get/Put/Delete example
- SpringBoot + Hibernate Spring JPA One-to-One Association + PostgreSQL | CRUD RestAPIs Post/Get/Put/Delete
- Angular 6 + Spring Boot + H2 Database [embedded mode] example | Spring Data JPA + RestAPIs CRUD example
- Angular 6 HttpClient + Spring Boot + MariaDB example | Spring Data JPA + RestAPIs CRUD example
- Spring Boot + Angular 6 example | Spring Data + REST + MongoDb CRUD example
- Spring Boot + Angular 6 example | Spring Data JPA + REST + MySQL CRUD example

## Post Tags

java datetime | jsonjackson | spring jpa | spring restapi | springboot mysql | springboot postgresql

## 2 thoughts on “Java Date Time – How to build SpringBoot RestApi – Post/Get request with Java Date Time using Jackson and Make Query with Spring JPA example”

**Venkatesh**

January 17, 2021 at 3:00 pm

Hi , Its an awesome blog on DAte and time but I am confused in repository section. can u explain how it is implemented .
i need further explanation in those 2 approaches used in DATETimeReposirory!

Waiting for your reply!

**Venkatesh**

January 17, 2021 at 4:55 pm

Also u havent mention about SpringBootRestApiDateTimeMySqlApplicationTests