

Difference between JPA, Hibernate and Spring Data JPA

FROM FILE: 1. spring-data-jpa-handson

JPA: is a specification for persisting Java objects in databases.

Hibernate : is an ORM tool that implements JPA and handles object-database mapping.

Spring Data JPA : simplifies JPA by reducing boilerplate code and providing built-in repository support.

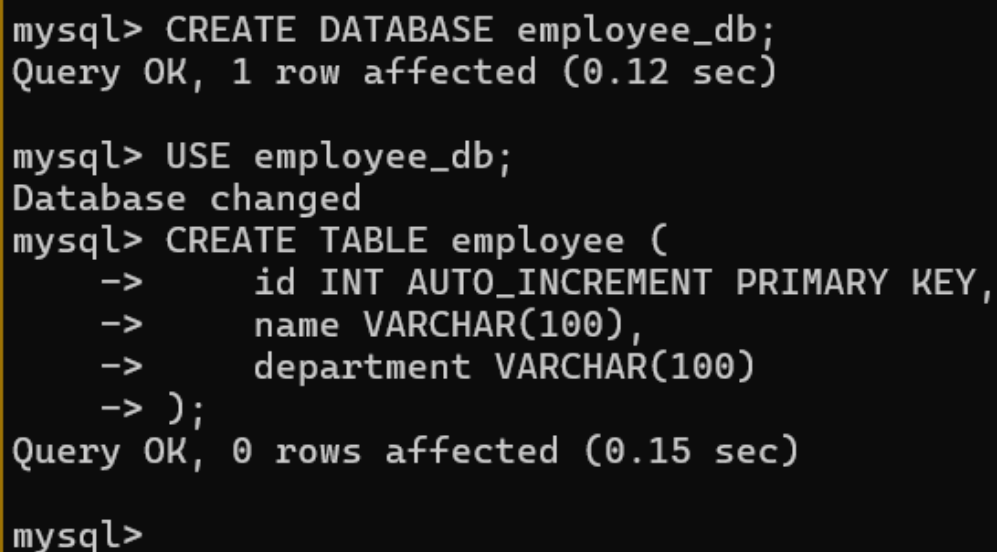
SQL commands for creating database:

```
CREATE DATABASE employee_db;
```

```
USE employee_db;
```

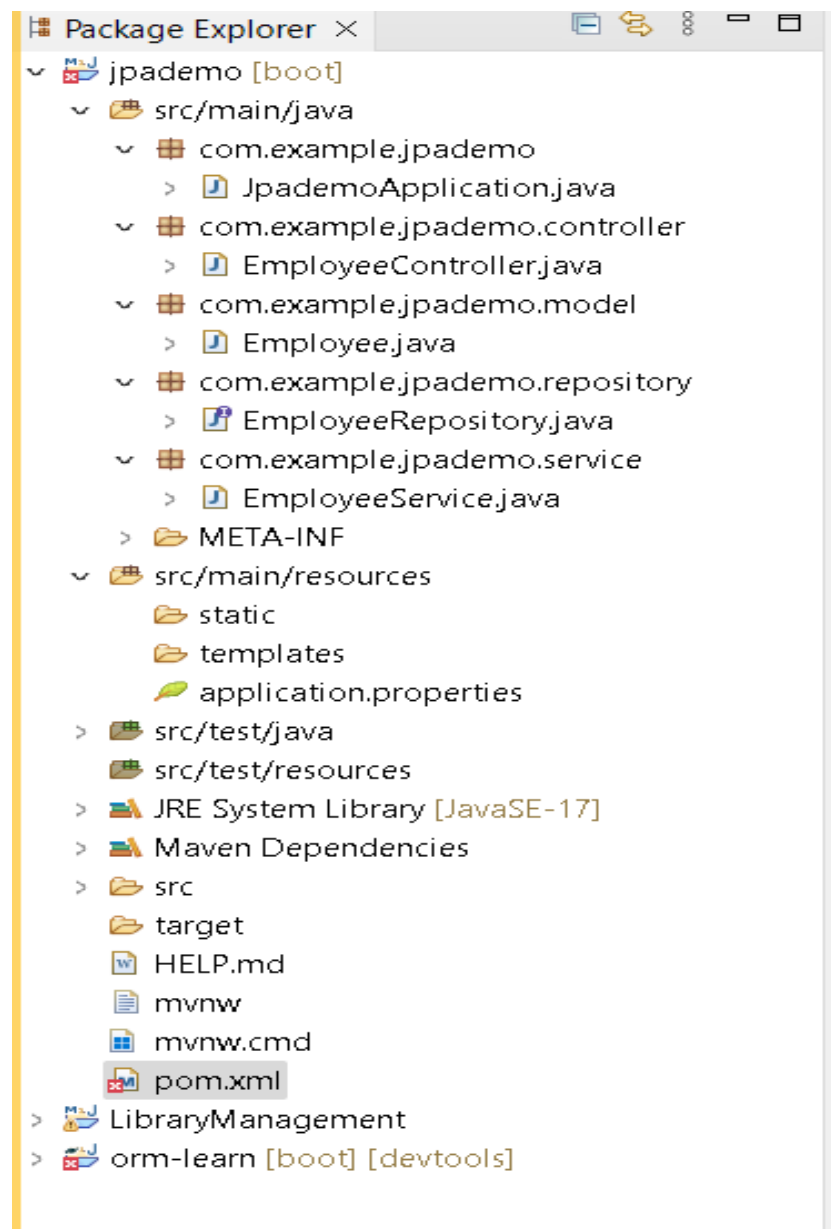
```
CREATE TABLE employee (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    name VARCHAR(100),  
    department VARCHAR(100)  
);
```

Pic From Command Prompt:



```
mysql> CREATE DATABASE employee_db;  
Query OK, 1 row affected (0.12 sec)  
  
mysql> USE employee_db;  
Database changed  
mysql> CREATE TABLE employee (  
    ->     id INT AUTO_INCREMENT PRIMARY KEY,  
    ->     name VARCHAR(100),  
    ->     department VARCHAR(100)  
    -> );  
Query OK, 0 rows affected (0.15 sec)  
  
mysql>
```

ProjectCreation:



JpademoApplication.java :

Code:

```
package com.example.jpademo;  
  
import org.springframework.boot.SpringApplication;  
  
import  
org.springframework.boot.autoconfigure.SpringBootApplication;
```

```

import
org.springframework.boot.autoconfigure.security.servlet.SecurityAut
oConfiguration;

@SpringBootApplication(exclude = { SecurityAutoConfiguration.class
})

public class JpademoApplication {

    public static void main(String[] args) {

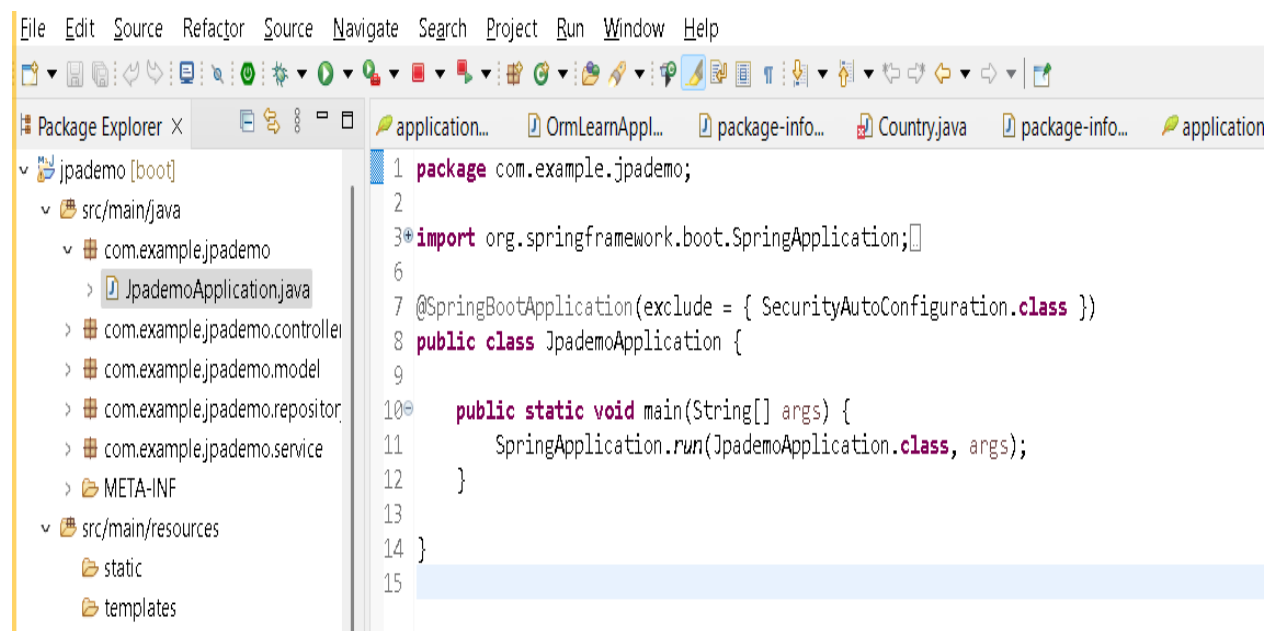
        SpringApplication.run(JpademoApplication.class, args);

    }

}

```

Pic From SpringTool :



EmployeeController.java

Code:

```
package com.example.jpdemo.controller;

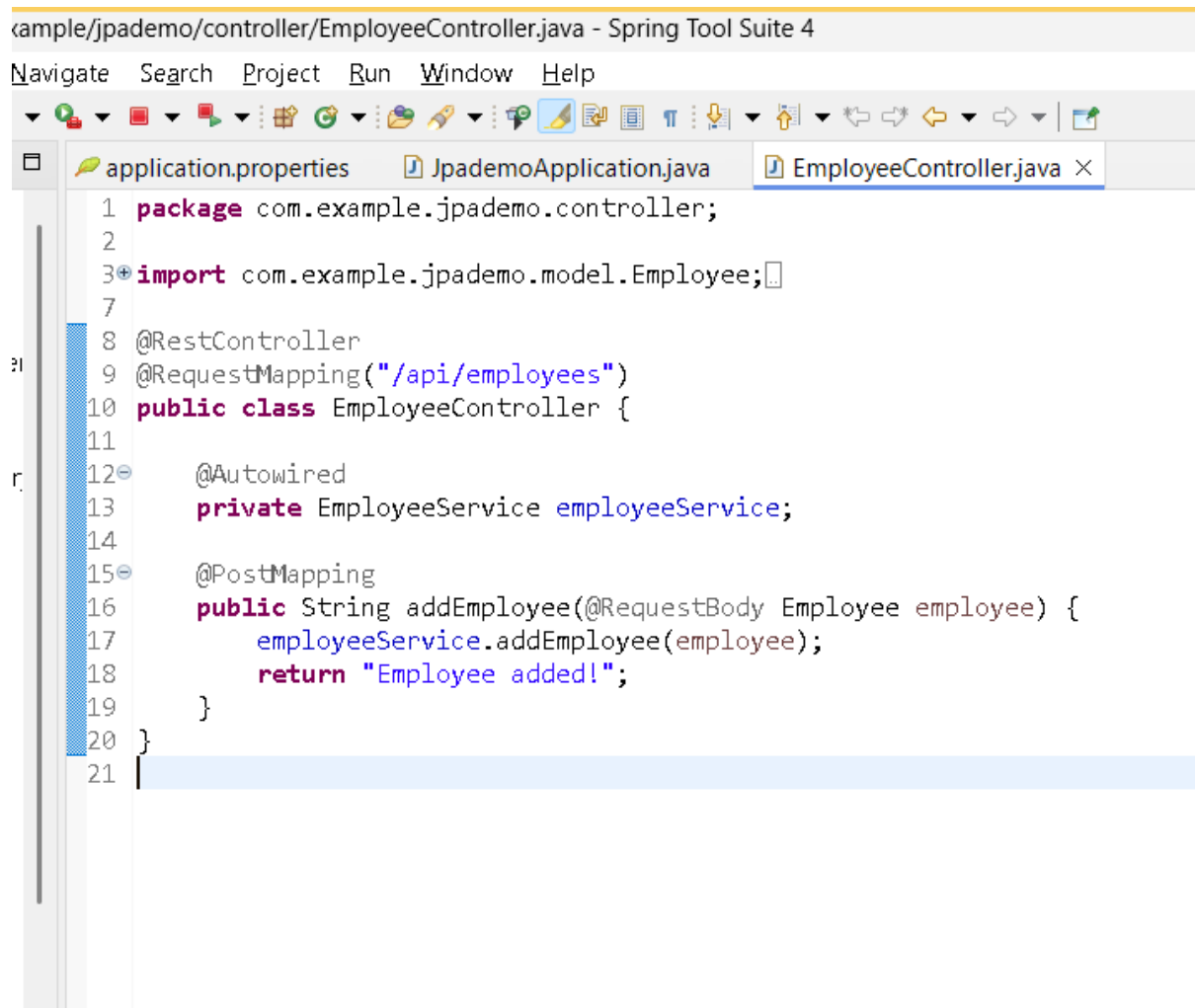
import com.example.jpdemo.model.Employee;
import com.example.jpdemo.service.EmployeeService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

@RestController
@RequestMapping("/api/employees")
public class EmployeeController {

    @Autowired
    private EmployeeService employeeService;

    @PostMapping
    public String addEmployee(@RequestBody Employee employee) {
        employeeService.addEmployee(employee);
        return "Employee added!";
    }
}
```

Pic From SpringTool:

A screenshot of the Spring Tool Suite 4 IDE. The title bar shows the file path: 'xample/jpdemo/controller/EmployeeController.java - Spring Tool Suite 4'. The menu bar includes 'Navigate', 'Search', 'Project', 'Run', 'Window', and 'Help'. Below the menu is a toolbar with various icons. The editor window shows the 'EmployeeController.java' file with the following code:

```
1 package com.example.jpdemo.controller;
2
3 import com.example.jpdemo.model.Employee;
4
5
6
7
8 @RestController
9 @RequestMapping("/api/employees")
10 public class EmployeeController {
11
12     @Autowired
13     private EmployeeService employeeService;
14
15     @PostMapping
16     public String addEmployee(@RequestBody Employee employee) {
17         employeeService.addEmployee(employee);
18         return "Employee added!";
19     }
20 }
21
```

Employee.java:

Code:

```
package com.example.jpdemo.model;

import jakarta.persistence.*;
```

@Entity

```
public class Employee {
```

```
    @Id
```

```
@GeneratedValue(strategy = GenerationType.IDENTITY)

private Integer id;

private String name;

private String department;

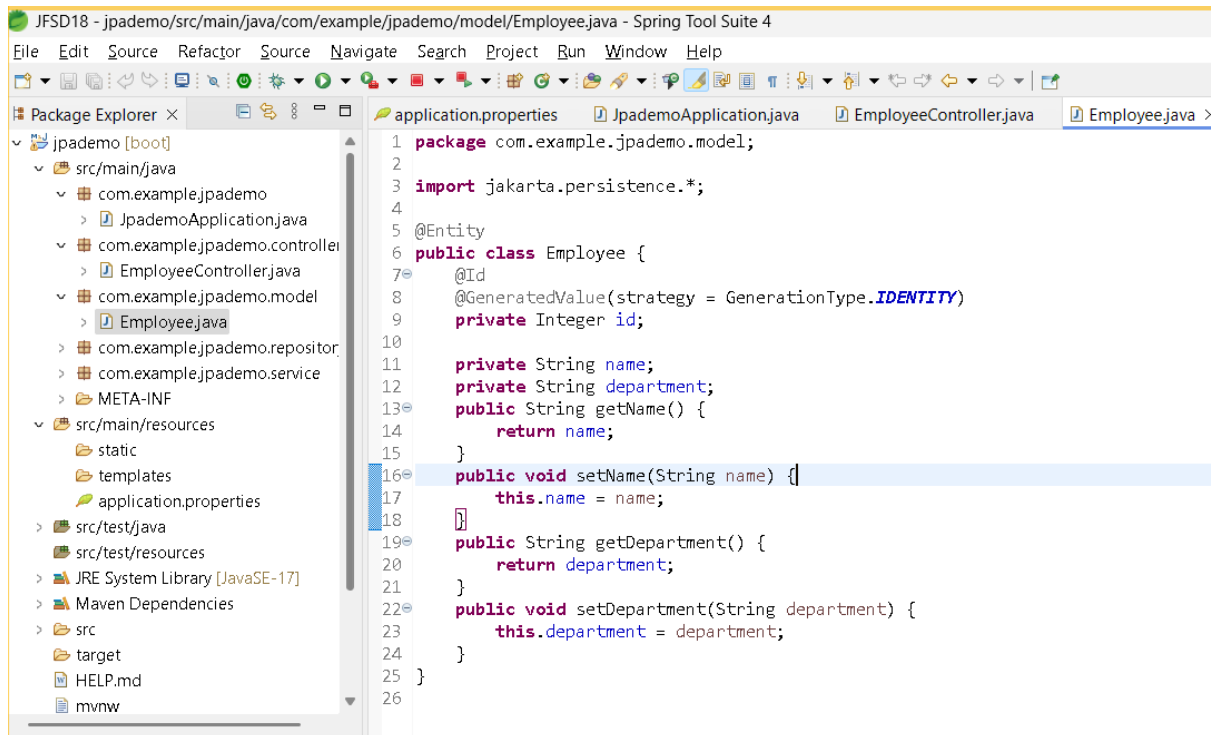

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getDepartment() {
    return department;
}

public void setDepartment(String department) {
    this.department = department;
}
}
```

Pic From SpringTool:



EmployeeRepository.java:

Code:

```
package com.example.jpademo.repository;
```

```
import com.example.jpademo.model.Employee;
```

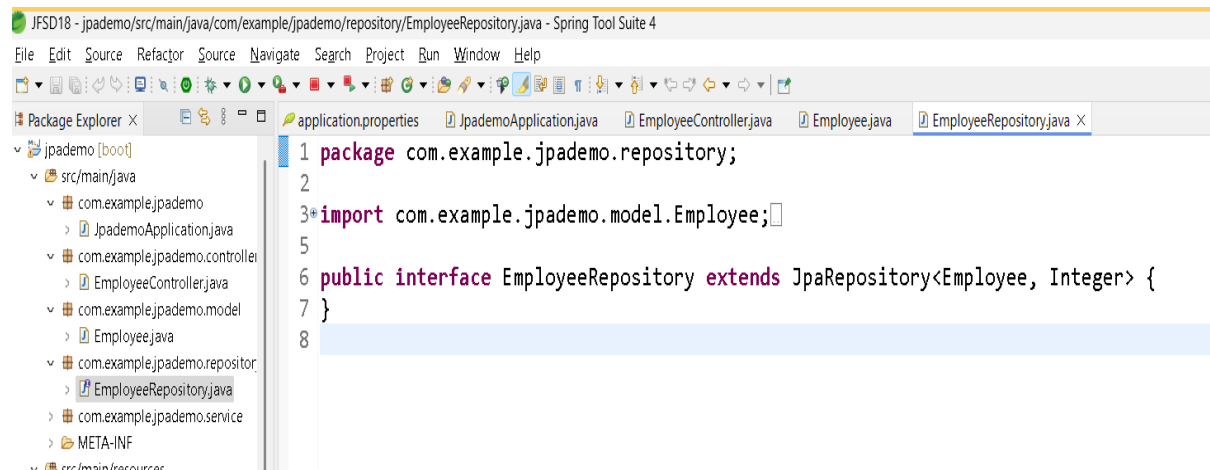
```
import org.springframework.data.jpa.repository.JpaRepository;
```

```
public interface EmployeeRepository extends
```

```
JpaRepository<Employee, Integer> {
```

```
}
```

Pic From SpringTool:



EmployeeService.java:

Code:

```
package com.example.jpademo.service;

import com.example.jpademo.model.Employee;
import com.example.jpademo.repository.EmployeeRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import jakarta.transaction.Transactional;
```

@Service

```
public class EmployeeService {
```

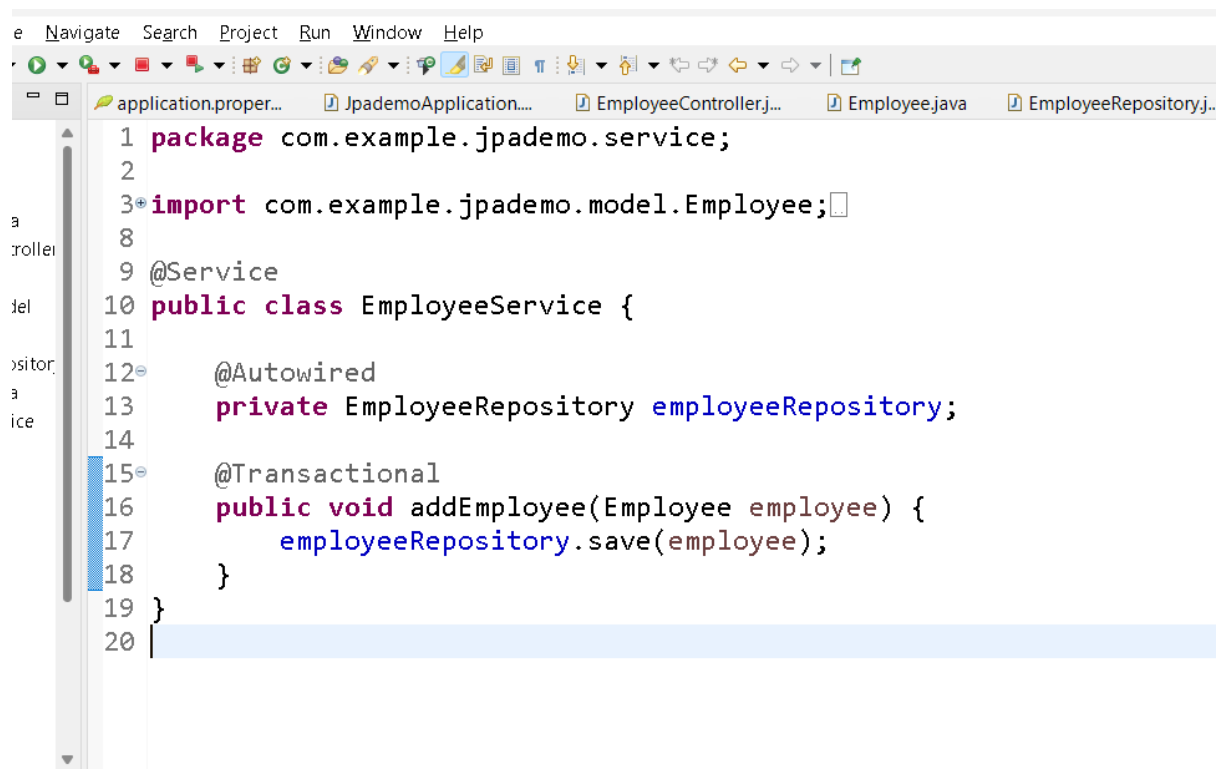
@Autowired

```
private EmployeeRepository employeeRepository;
```


@Transactional

```
public void addEmployee(Employee employee) {  
    employeeRepository.save(employee);  
}  
}
```

Pic From SpringTool:



Application.properties code:

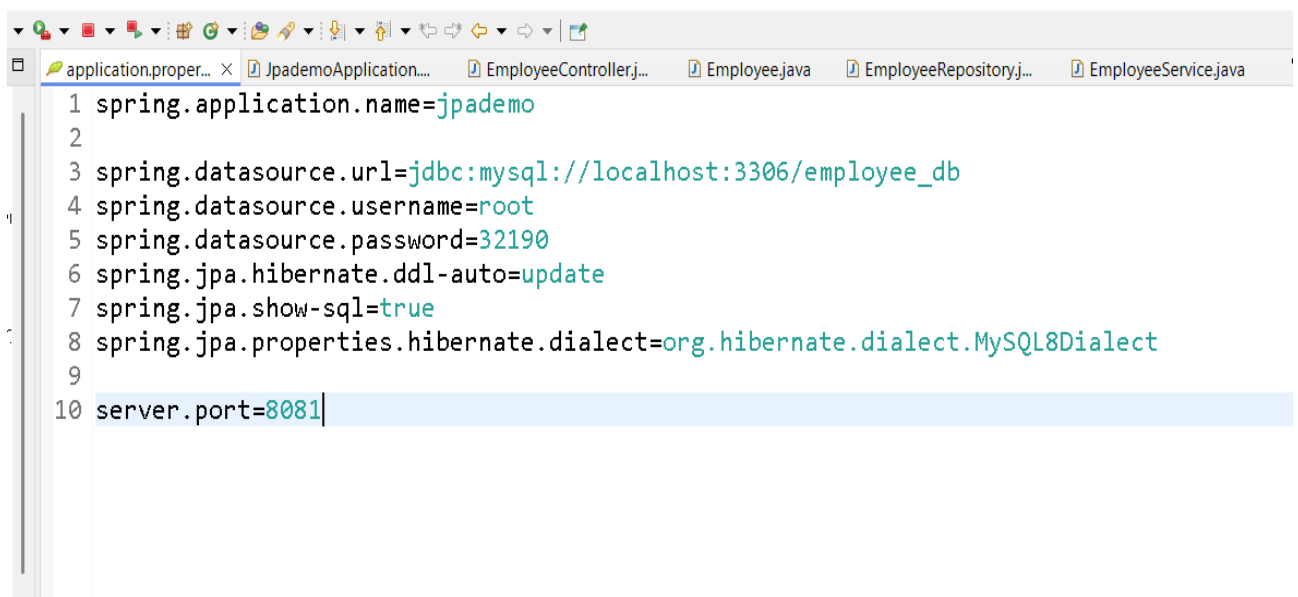
```
spring.application.name=jpademo  
spring.datasource.url=jdbc:mysql://localhost:3306/employee_db  
spring.datasource.username=root  
spring.datasource.password=32190  
spring.jpa.hibernate.ddl-auto=update
```

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect

server.port=8081

PIC From SpringTool:

A screenshot of an IDE window showing the configuration properties for a Spring application. The window has several tabs at the top: 'application.properties', 'JpademoApplication...', 'EmployeeController.j...', 'Employee.java', 'EmployeeRepository.j...', and 'EmployeeService.java'. The 'application.properties' tab is active, displaying a list of properties. The properties are numbered 1 through 10. The 10th property, 'server.port=8081', is highlighted with a blue background. The other properties are: 1. spring.application.name=jpademo, 2. (empty), 3. spring.datasource.url=jdbc:mysql://localhost:3306/employee_db, 4. spring.datasource.username=root, 5. spring.datasource.password=32190, 6. spring.jpa.hibernate.ddl-auto=update, 7. spring.jpa.show-sql=true, 8. spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect, 9. (empty).

```
1 spring.application.name=jpademo
2
3 spring.datasource.url=jdbc:mysql://localhost:3306/employee_db
4 spring.datasource.username=root
5 spring.datasource.password=32190
6 spring.jpa.hibernate.ddl-auto=update
7 spring.jpa.show-sql=true
8 spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
9
10 server.port=8081
```

Pom.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
```

```
    https://maven.apache.org/xsd/maven-4.0.0.xsd>
```

```
    <modelVersion>4.0.0</modelVersion>
```

```
    <parent>
```

```
        <groupId>org.springframework.boot</groupId>
```

```
        <artifactId>spring-boot-starter-parent</artifactId>
```

```
<version>3.5.3</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.example</groupId>

<artifactId>jpademo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>jpademo</name>

<description>Demo project for Spring Boot</description>

<url/>

<licenses>

    <license/>

</licenses>

<developers>

    <developer/>

</developers>

<scm>

    <connection/>

    <developerConnection/>

    <tag/>

    <url/>

</scm>

<properties>

    <java.version>17</java.version>
```

</properties>

<dependencies>

 <dependency>

 <groupId>org.springframework.boot</groupId>

 <artifactId>spring-boot-starter-data-jpa</artifactId>

 </dependency>

 <dependency>

 <groupId>org.springframework.boot</groupId>

 <artifactId>spring-boot-starter-web</artifactId>

 </dependency>

 <dependency>

 <groupId>com.mysql</groupId>

 <artifactId>mysql-connector-j</artifactId>

 <scope>runtime</scope>

 </dependency>

 <dependency>

 <groupId>org.springframework.boot</groupId>

 <artifactId>spring-boot-starter-test</artifactId>

 <scope>test</scope>

 </dependency>

</dependencies>

```

<build>

    <plugins>

        <plugin>

            <groupId>org.springframework.boot</groupId>

            <artifactId>spring-boot-maven-
plugin</artifactId>

        </plugin>

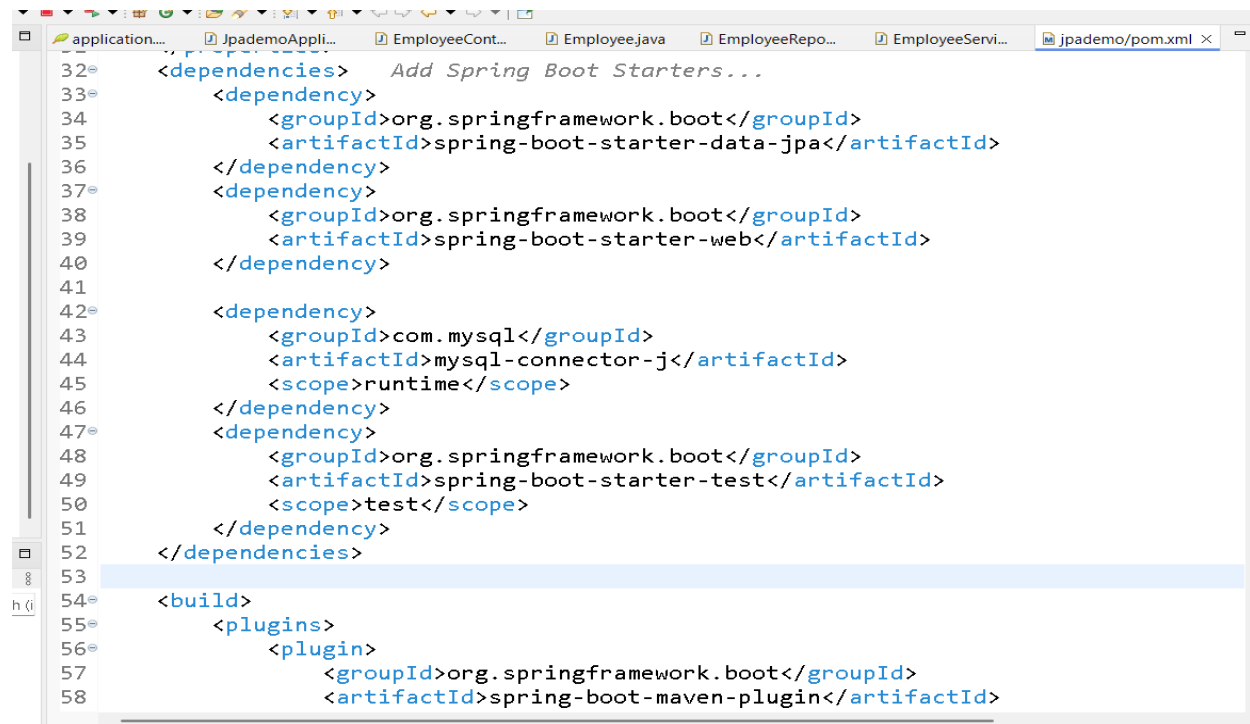
    </plugins>

</build>

</project>

```

Pic From SpringTool:



```

32<dependencies>    Add Spring Boot Starters...
33<dependency>
34    <groupId>org.springframework.boot</groupId>
35    <artifactId>spring-boot-starter-data-jpa</artifactId>
36</dependency>
37<dependency>
38    <groupId>org.springframework.boot</groupId>
39    <artifactId>spring-boot-starter-web</artifactId>
40</dependency>
41
42<dependency>
43    <groupId>com.mysql</groupId>
44    <artifactId>mysql-connector-j</artifactId>
45    <scope>runtime</scope>
46</dependency>
47<dependency>
48    <groupId>org.springframework.boot</groupId>
49    <artifactId>spring-boot-starter-test</artifactId>
50    <scope>test</scope>
51</dependency>
52</dependencies>
53
54<build>
55    <plugins>
56        <plugin>
57            <groupId>org.springframework.boot</groupId>
58            <artifactId>spring-boot-maven-plugin</artifactId>

```

Output After Running SpringBoot App:

```
Javadoc Declaration Console X
jpademo - JpademoApplication [Spring Boot App] C:\Users\91833\Downloads\sts-4.23.1.RELEASE\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_21.0.3.v20240426-1530\jre\bin\javaw.exe (05-Jul-2025, 11:56:50 pm) [pid
2025-07-05T23:56:56.342+05:30 INFO 7356 --- [jpademo] [main] com.zaxxer.hikari.pool.HikariPool : HikariPool-1 - Added connection com.mysql.cj.jdbc.Conne
2025-07-05T23:56:56.344+05:30 INFO 7356 --- [jpademo] [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2025-07-05T23:56:56.435+05:30 WARN 7356 --- [jpademo] [main] org.hibernate.orm.deprecation : HHH90000025: MySQLDialect does not need to be specifie
2025-07-05T23:56:56.436+05:30 WARN 7356 --- [jpademo] [main] org.hibernate.orm.deprecation : HHH90000026: MySQLDialect has been deprecated; use org
2025-07-05T23:56:56.462+05:30 INFO 7356 --- [jpademo] [main] org.hibernate.orm.connections.pooling : HHH10001005: Database info:
Database JDBC URL [Connecting through datasource 'HikariDataSource (HikariPool-1)']
Database driver: undefined/unknown
Database version: 8.0
Autocommit mode: undefined/unknown
Isolation level: undefined/unknown
Minimum pool size: undefined/unknown
Maximum pool size: undefined/unknown
2025-07-05T23:56:57.417+05:30 INFO 7356 --- [jpademo] [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000489: No JTA platform available (set 'hibernate.tr
Hibernate: alter table employee modify column department varchar(255)
Hibernate: alter table employee modify column name varchar(255)
2025-07-05T23:56:57.753+05:30 INFO 7356 --- [jpademo] [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence un
2025-07-05T23:56:58.081+05:30 WARN 7356 --- [jpademo] [main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefor
2025-07-05T23:56:58.496+05:30 INFO 7356 --- [jpademo] [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8081 (http) with context path '/'
2025-07-05T23:56:58.506+05:30 INFO 7356 --- [jpademo] [main] com.example.jpademo.JpademoApplication : Started JpademoApplication in 5.89 seconds (process runi
2025-07-06T00:05:47.307+05:30 INFO 7356 --- [jpademo] [nio-8081-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'
2025-07-06T00:05:47.315+05:30 INFO 7356 --- [jpademo] [nio-8081-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2025-07-06T00:05:47.352+05:30 INFO 7356 --- [jpademo] [nio-8081-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 35 ms
Hibernate: insert into employee (department,name) values (?,?)
```

Search Postman

rt

POST http://localhost:8081/ap

+

...

oo

HTTP

http://localhost:8081/api/employees

POST

▼

http://localhost:8081/api/employees

Params

Authorization

Headers (8)

Body ●

Pre-request Script

Tests

Settings

● none

● form-data

● x-www-form-urlencoded

● raw

● binary

JSON ▼

1 {

2 "name": "Sruthi",

3 "department": "BackendDev"

4 }

5

Body

Cookies

Headers (5)

Test Results

Pretty

Raw

Preview

Visualize

Text ▼

↻

1 Employee added!