This is different tutorial..

Steps to Configure Hibernate in eclipse

Hibernate
Installatation Guide

Steps to Configure Hibernate in eclipse

Please refer hibernate tutorial – give above link

Creating entity class in eclipse

```
package com.jbk;
// Generated Jan 17, 2020 12:47:15 AM by Hibernate Tools 5.2.10.Final

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import static javax.persistence.GenerationType.IDENTITY;
import javax.persistence.Id;
import javax.persistence.Table;

/**
 * Employee generated by hbm2java
 */
@Entity
@Table(name = "employee", catalog = "test")
public class Employee implements java.io.Serializable {

	private Integer eid;
	private String ename;

	public Employee() {
	}

	public Employee(String ename) {
		this.ename = ename;
	}

	@Id
	@GeneratedValue(strategy = IDENTITY)
	@Column(name = "eid", unique = true, nullable = false)
	public Integer getEid() {
		return this.eid;
	}

	public void setEid(Integer eid) {
		this.eid = eid;
	}

	@Column(name = "ename", nullable = false, length = 45)
	public String getEname() {
		return this.ename;
	}

	public void setEname(String ename) {
		this.ename = ename;
	}

}
```

application.properties

```
## Spring DATASOURCE (DataSourceAutoConfiguration & DataSourceProperties)
spring.datasource.url = jdbc:mysql://localhost:3306/test
spring.datasource.username = root
spring.datasource.password = root
spring.datasource.driver-class-name=com.mysql.jdbc.Driver


## Hibernate Properties
# The SQL dialect makes Hibernate generate better SQL for the chosen database
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect

# Hibernate ddl auto (create, create-drop, validate, update)
#spring.jpa.hibernate.ddl-auto = update
```

Complete flow Spring boot

Configuration for Datasource and SessionFactory

```
package com.jbk;

import javax.sql.DataSource;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.orm.hibernate5.LocalSessionFactoryBean;

@Configuration
public class Config {
	@Autowired
	DataSource dataSource;

	@Bean
	public LocalSessionFactoryBean sessionFactory() {
		LocalSessionFactoryBean sessionFactory = new LocalSessionFactoryBean();
		sessionFactory.setDataSource(dataSource);
		sessionFactory.setAnnotatedClasses(Employee.class);
		return sessionFactory;
	}
}
```

Pom.xml dependencies

```
<?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>2.2.2.RELEASE</version>
		<relativePath /> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.jbk</groupId>
	<artifactId>demospringbootjbk</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>demospringbootjbk</name>
	<description>Demo project for Spring Boot</description>

	<properties>
		<java.version>1.8</java.version>
		<maven-jar-plugin.version>3.1.1</maven-jar-plugin.version>
	</properties>

	<dependencies>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-data-rest</artifactId>
		</dependency>


		<!-- Hibernate integration -->
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-data-jpa</artifactId>
		</dependency>
		<dependency>
			<groupId>mysql</groupId>
			<artifactId>mysql-connector-java</artifactId>
			<version>5.1.6</version>
		</dependency>

	</dependencies>

	<build>
		<plugins>
			<plugin>
				<groupId>org.springframework.boot</groupId>
				<artifactId>spring-boot-maven-plugin</artifactId>
			</plugin>
		</plugins>
	</build>

</project>
```

Controller class

```
package com.jbk;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequestMapping("api")
public class EmployeeController {
	@Autowired
	EmployeeService empService;

	@RequestMapping("namesData")
	String[] giveYourNames() {
		System.out.println(empService);
		return empService.giveYourNames();
	}
}
```

Service class

```
package com.jbk;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class EmployeeService {
	@Autowired
	EmployeeDao employeeDao;

	public String[] giveYourNames() {
		return employeeDao.giveYourNames();
	}

}
```

Dao Class

```
package com.jbk;

import org.hibernate.SessionFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;

```

```
@Repository

public class EmployeeDao {


	@Autowired
	private SessionFactory sessionFactory;

	String[] giveYourNames() {// db call
		System.out.println("sessionFactory >>>"+sessionFactory);
		Employee
employee=(Employee)sessionFactory.getCurrentSession().load(Employee.class, 1);
		String xx[] = { employee.getEname(), employee.getEname()+"javabykiran"
};
		return xx;
	}



}
```