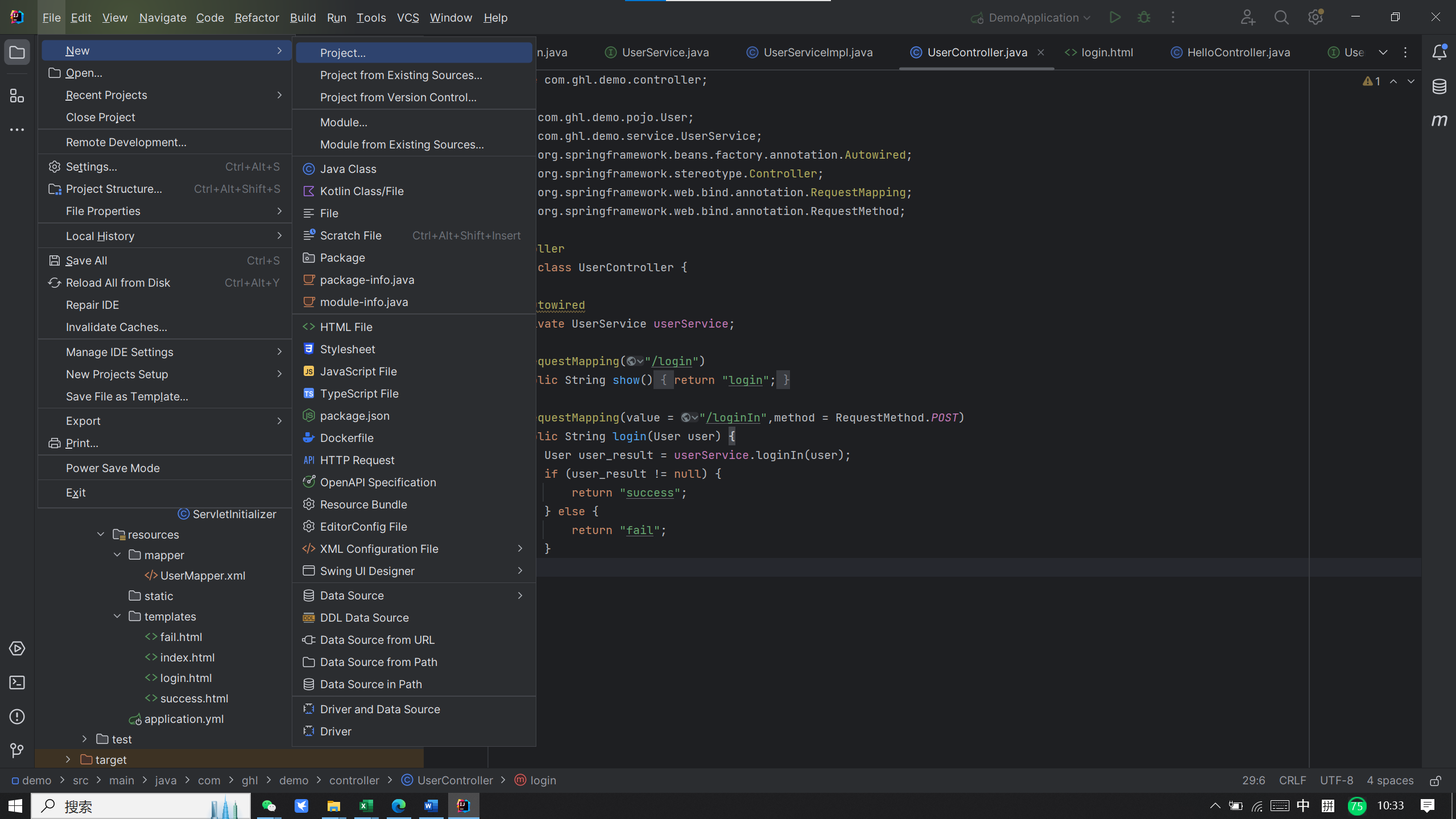
**使用IDEA搭建一个简单的SpringBoot登录项目**

**一、创建项目**

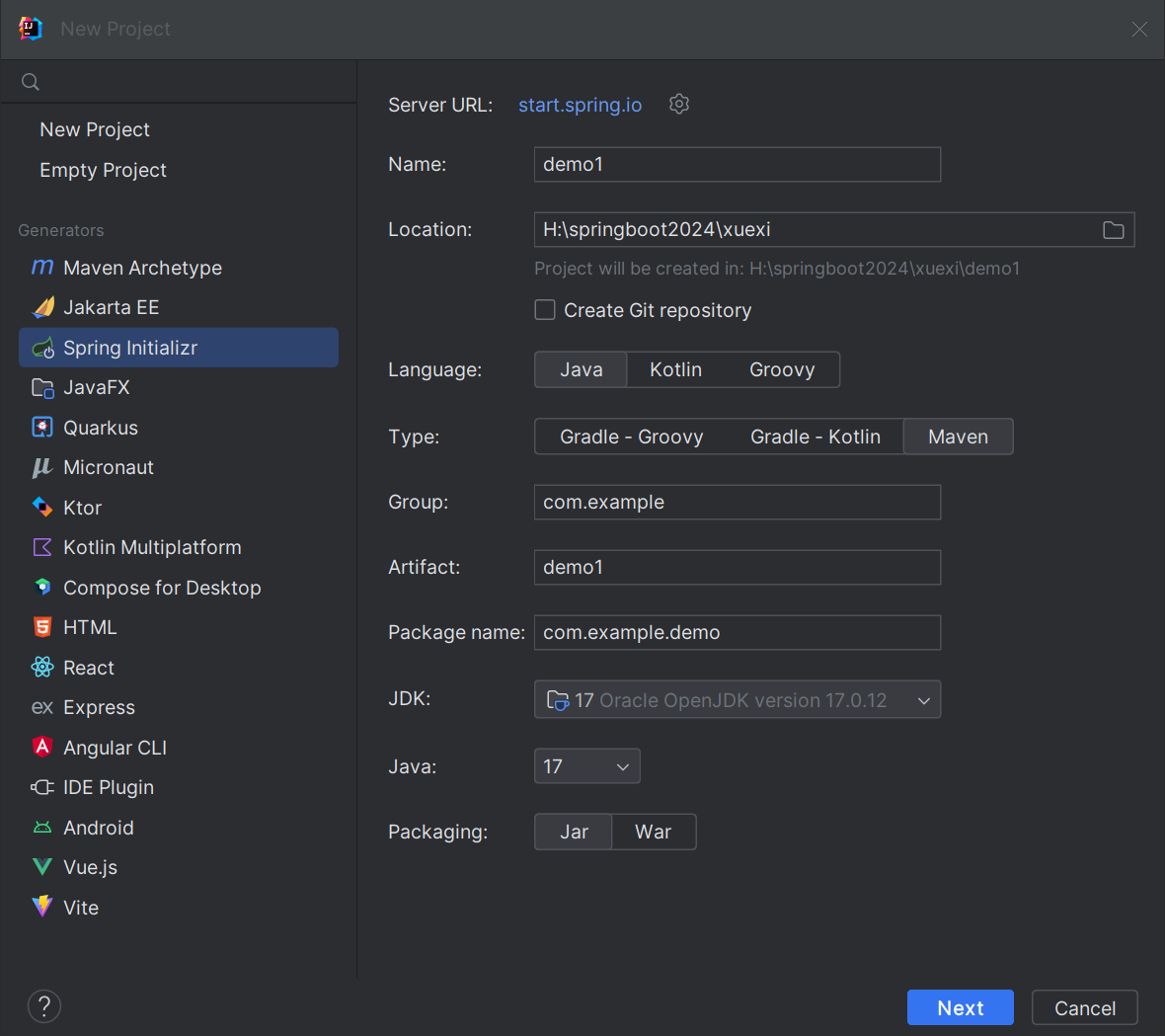
1.File->new->project；



2.选择“Spring Initializr”，点击next；（jdk17默认即可）



3.完善项目信息，组名可不做修改，项目名可做修改。最终建的项目名为：test，src->main->java下包名会是：com->ghl->test；点击next；

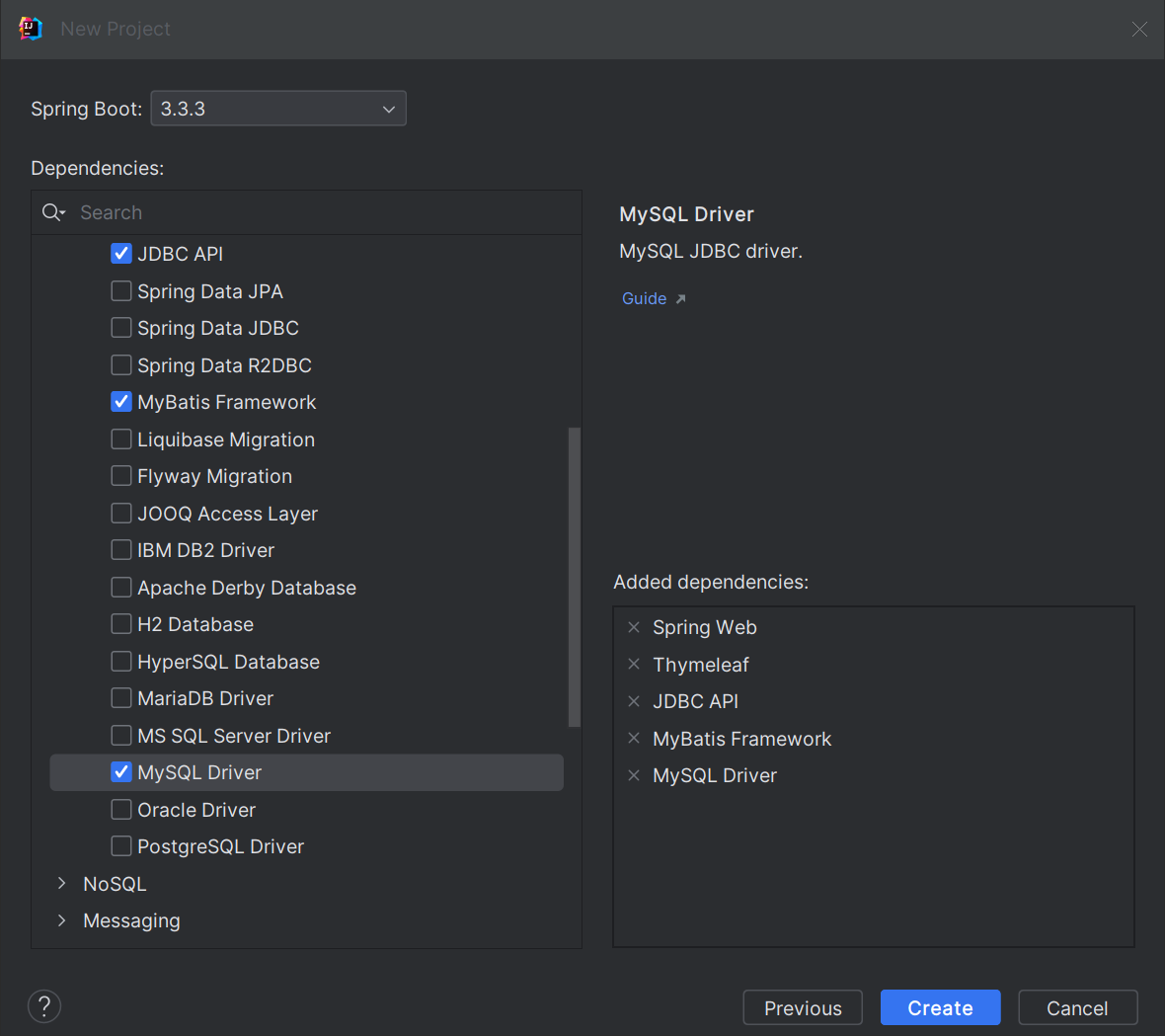


GroupID表示组织ID，一般分为两个字段，包括域名和公司名；ArtifactID表示项目唯一标识符；Version表示项目版本号。

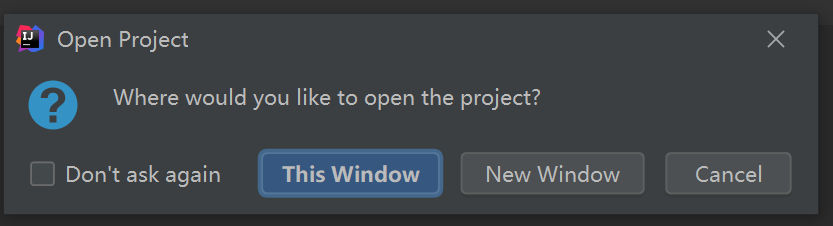
注意：默认打包为jar，如果要打war包，选择war。Type选择“Maven（General a Maven based project archive）”

4.Web下勾选Spring Web，（网上创建springboot项目多是勾选Web选项，而较高版本的SpringBoot没有此选项，勾选Spring Web即可，2.1.8版本是Spring Web）；Template Englines勾选Thymeleaf；SQL勾选：MySQL Driver，JDBC API和MyBatis Framework三项；点击【Create】；

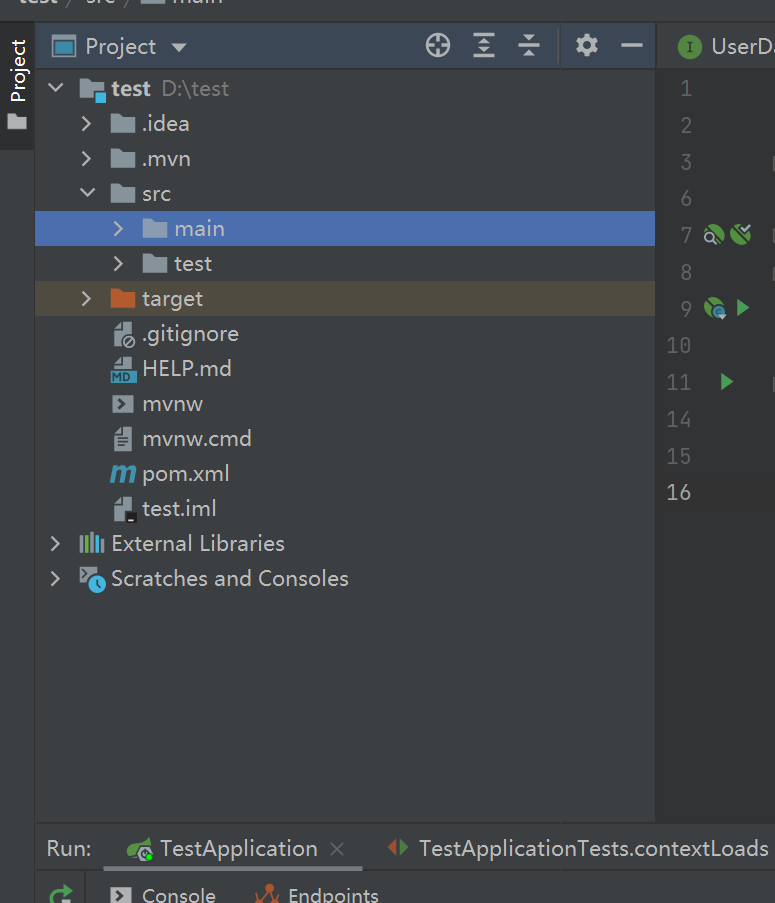
注意：如果springboot版本在3.0（包含3.0）以上，JDK必须是JDK17以上。



5.选择项目路径，点击【This Windows】；打开新的窗口；

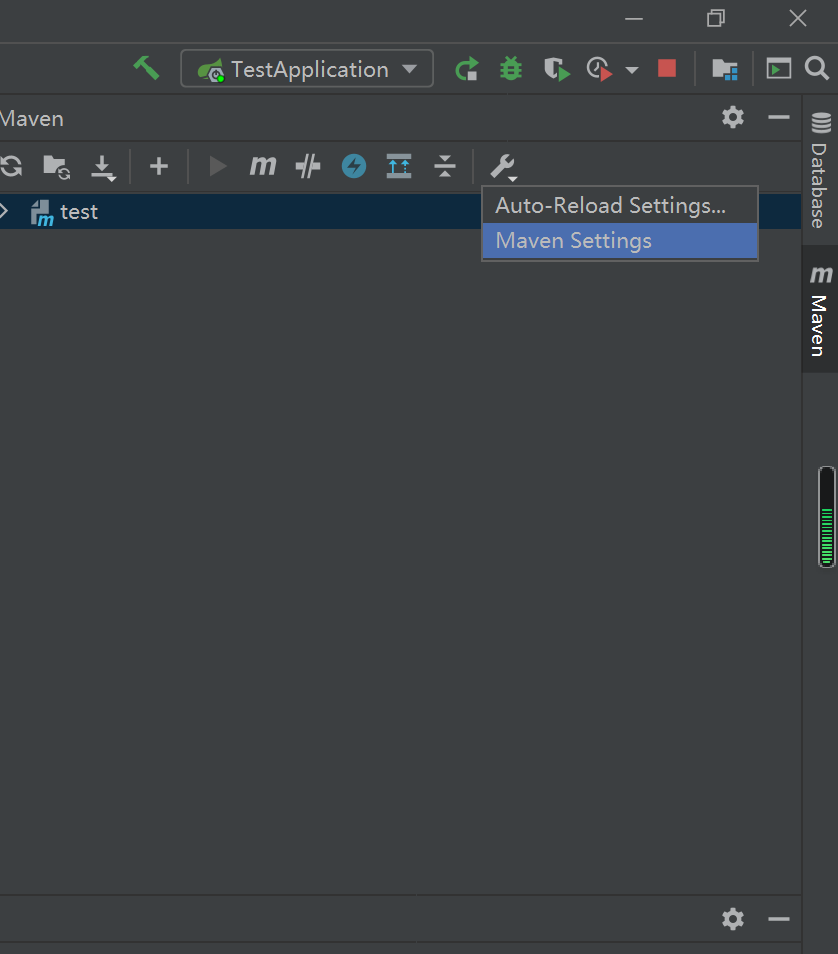


6.创建好的项目目录结构



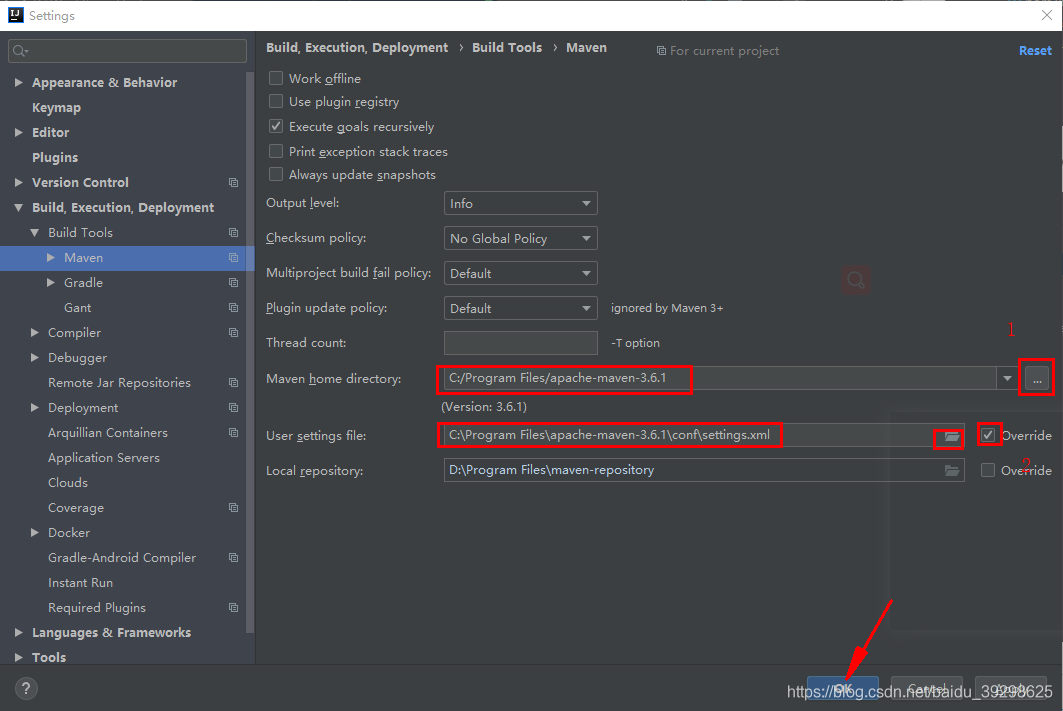
**二、Maven库设置**

7.点击右侧的Maven，点击设置（扳手图标）进行项目Maven仓库的配置；

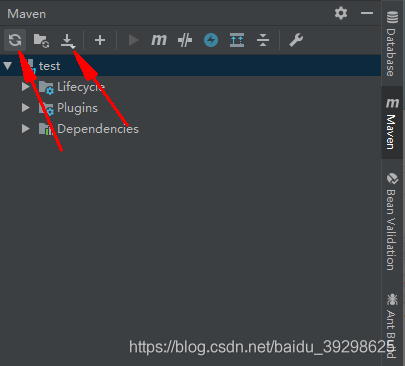


8.（1）选择本地Maven路径；

（2）勾选配置文件后边的选项，然后修改为本地Maven的配置文件，它会根据配置文件直接找到本地仓库位置；

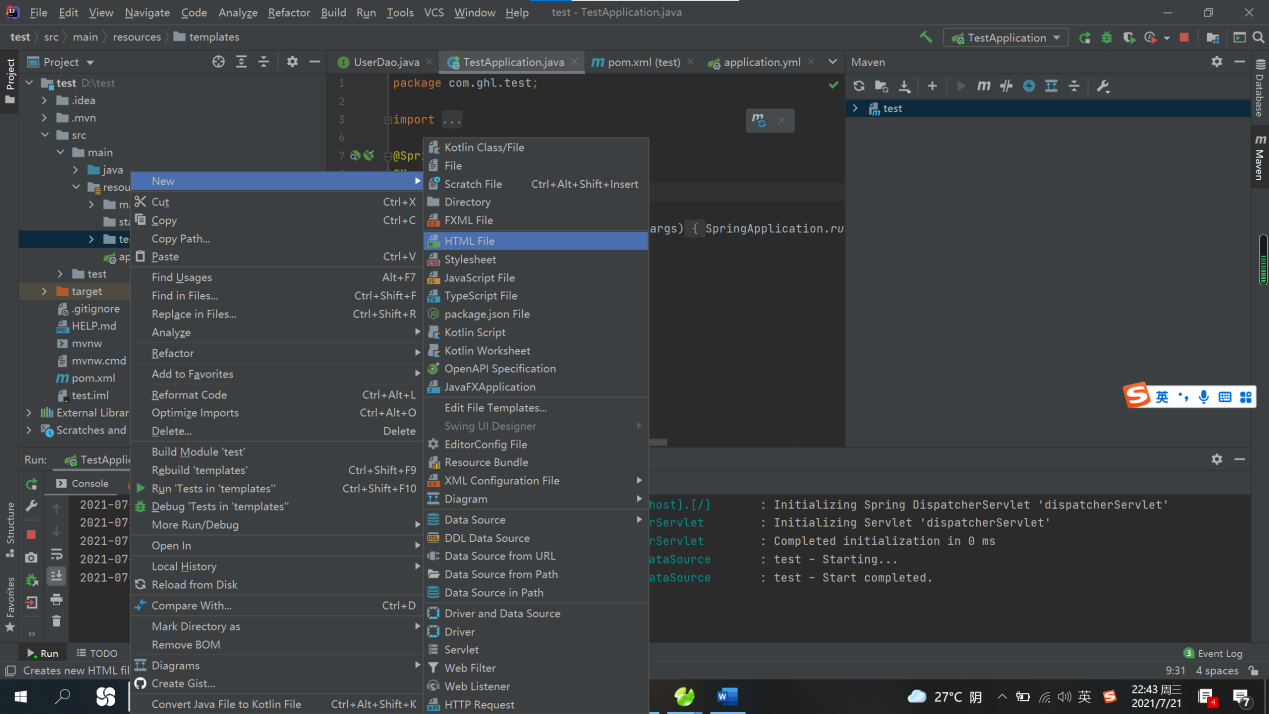


9.配置完后，如果没有自动导入包，可以点击左上角【刷新】导包按钮，或者按下【下载】按钮，选择下载所有源文件和文档



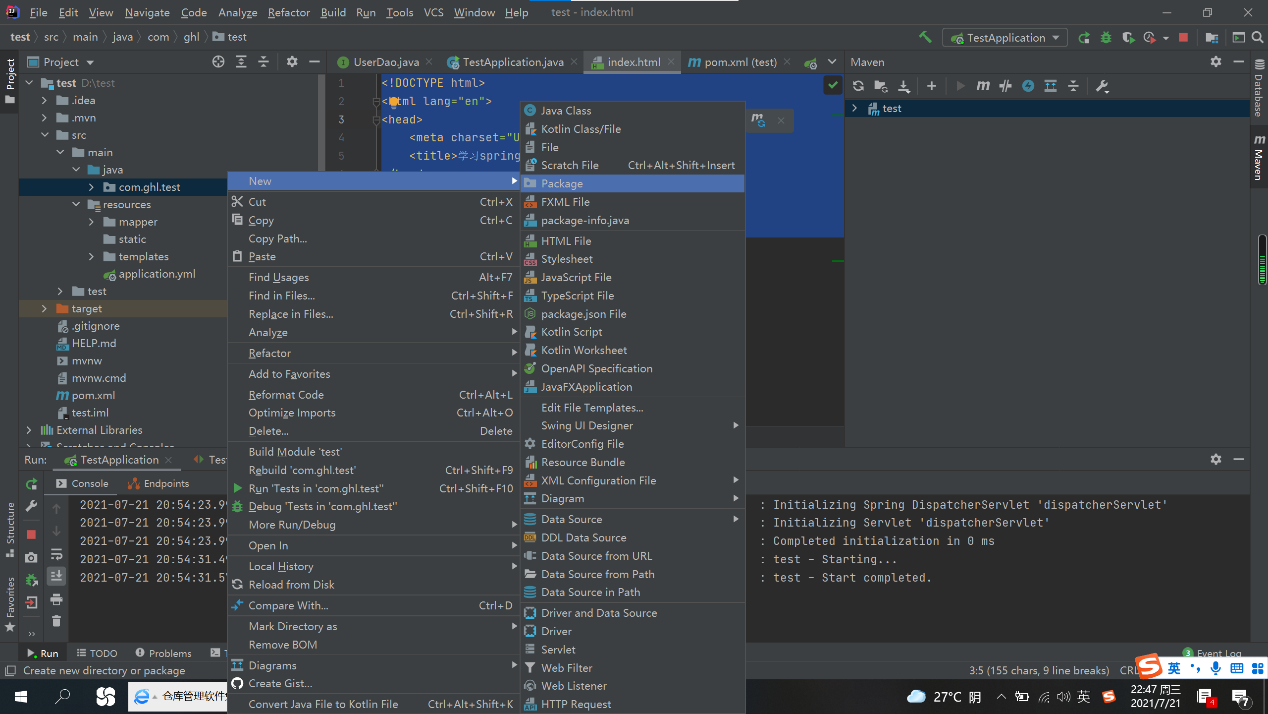
**三、测试项目**

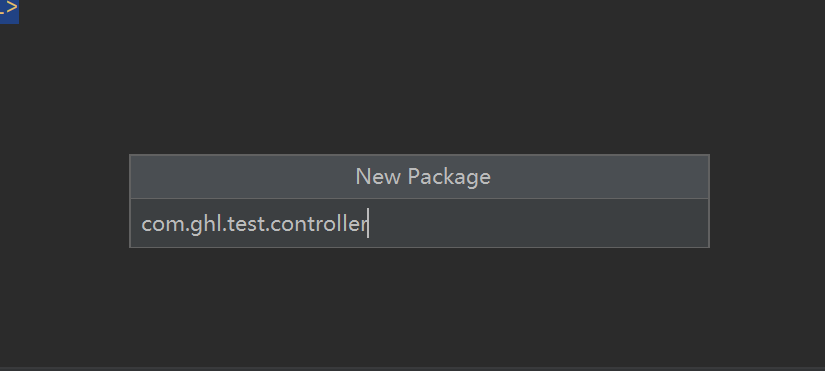
10.在templates文件夹下新建index.html页面，作为启动的初始页面；



<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>学习springboot</title>  
</head>  
<body>  
<h2>今天开始学习springboot</h2>  
</body>  
</html>

11.在com.ghl.test下新建包：controller，在controller文件夹下建一个简单的HelloController类；（Controller类要添加@Controller注解，项目启动时，SpringBoot会自动扫描加载Controller）。







package com.ghl.test.controller;

import org.springframework.stereotype.Controller;

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

@Controller

public class HelloController {

@RequestMapping("/index")

public String sayHello(){

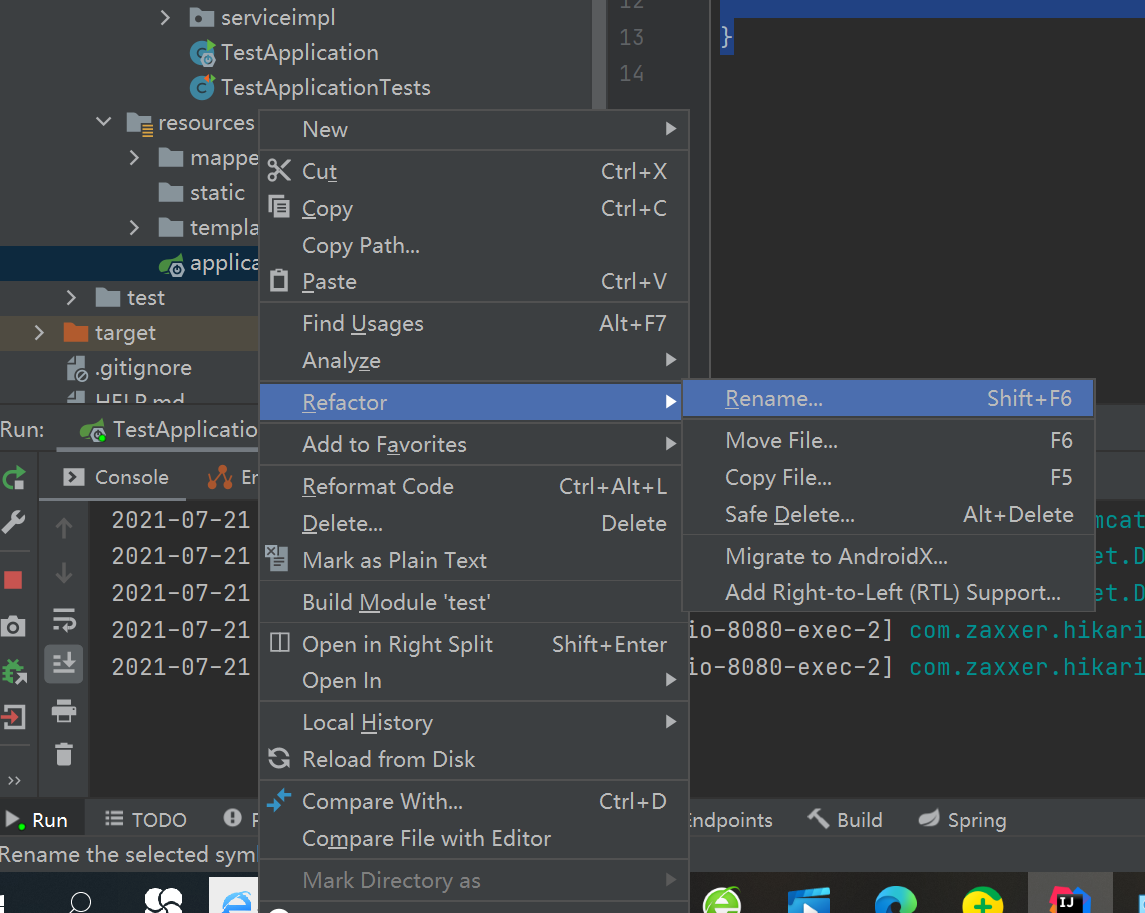
return "index";

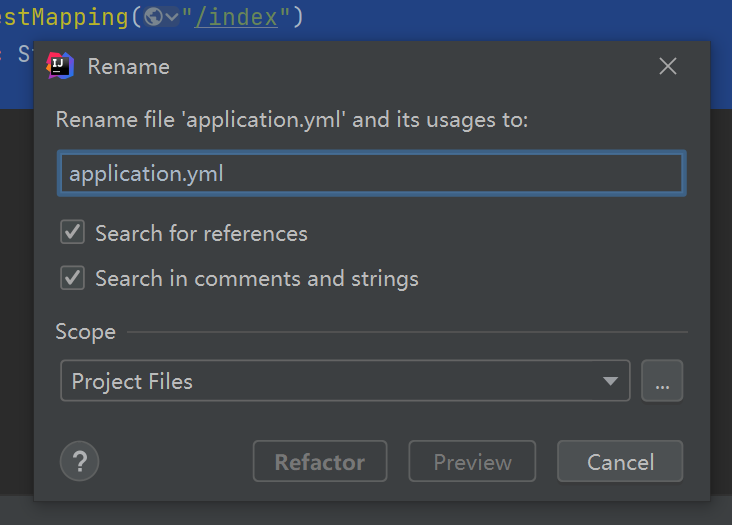
}

}

**四、系统配置文件设置**

12.在resources文件夹下application中先配置DataSource基本信息，application文件有两种文件格式，一种是以.properties为后缀（默认），一种是以.yml为后缀的，两种配置方式略有差别，详情可参考这个网址：https://blog.csdn.net/qq\_29648651/article/details/78503853；在这用.yml后缀的文件格式。右键application文件选择Refactor，选择Rename，将后缀改为yml；





spring:

datasource:

name: test #数据库名

url: jdbc:mysql://localhost:3306/test?useUnicode=true&characterEncoding=UTF-8 #url

username: root #用户名

password: #密码

driver-class-name: com.mysql.cj.jdbc.Driver #数据库连接驱动

注意：

1.如果MySQL数据库JDBC驱动是5.X，数据库连接字符串应改为：

driver-class-name: com.mysql.jdbc.Driver #数据库链接驱动

很多数据库连接失败就是这造成的。

2.mysql8.X的jdbc升级了，增加了时区（serverTimezone）属性，并且不允许为空。

解决方法：

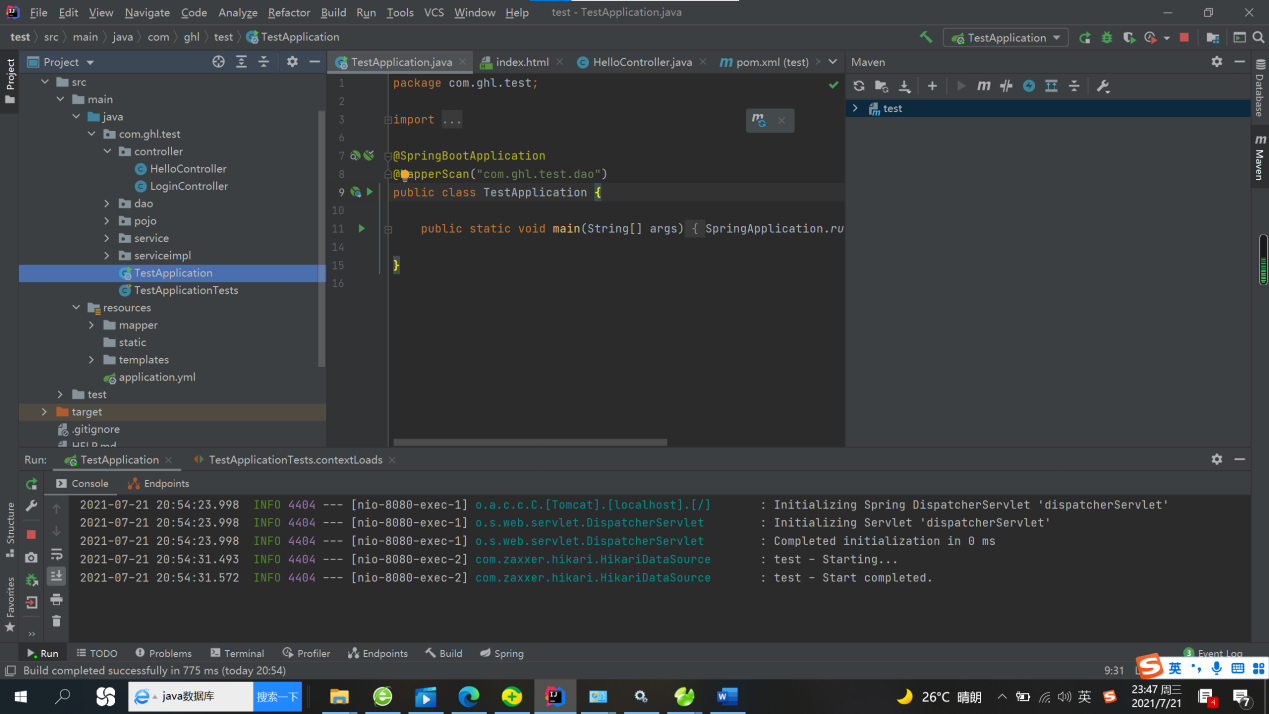
mysql的连接地址设置设置为亚洲上海

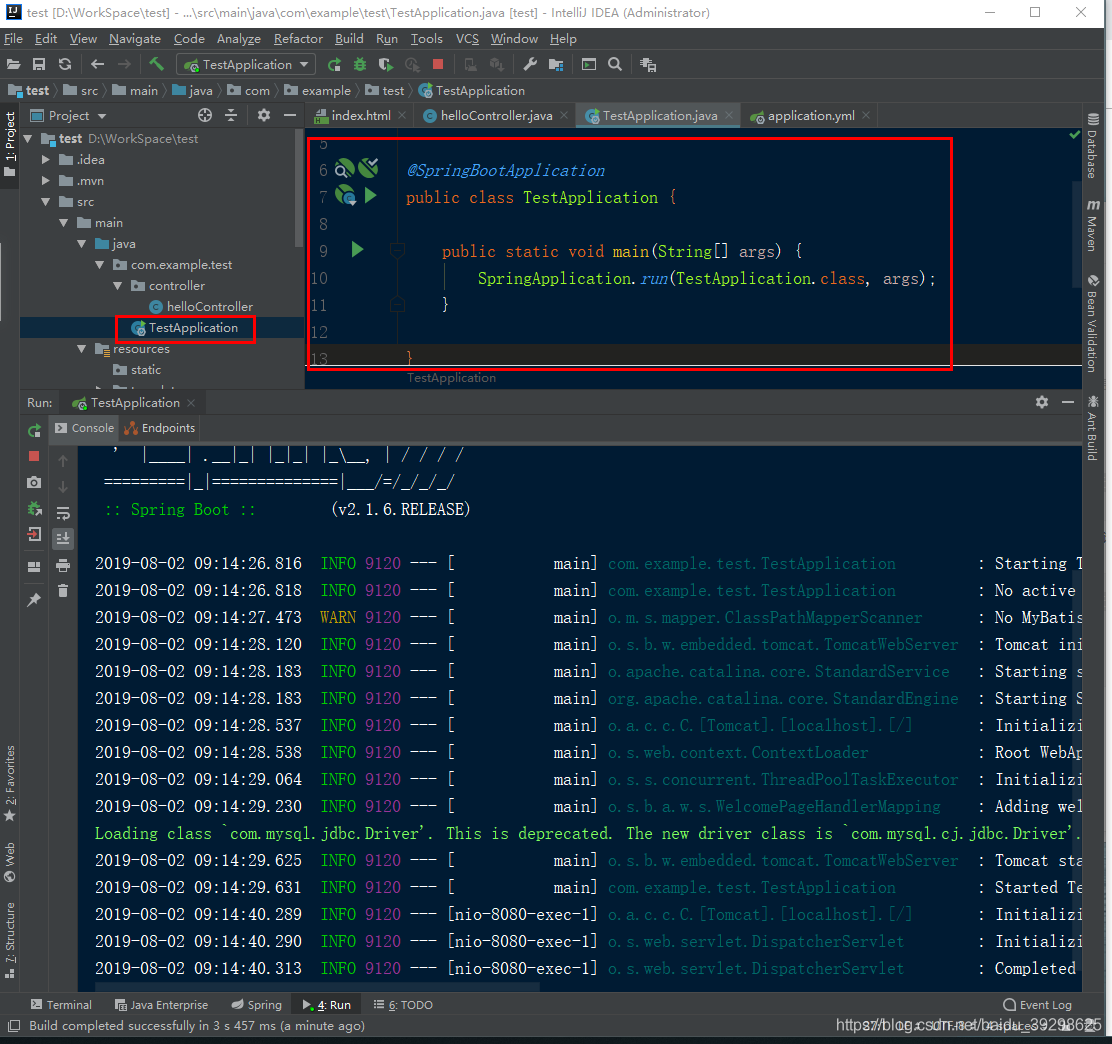
如果你是在本地运⾏代码，在JDBC url加上 &serverTimezone=Asia/Shanghai或者&serverTimezone=GMT%2B

url: jdbc:mysql://localhost:3306/firedb?useUnicode=true&characterEncoding=UTF-8&serverTimezone=Asia/Shanghai

**五、运行项目**

13.运行项目启动类TestApplication.java

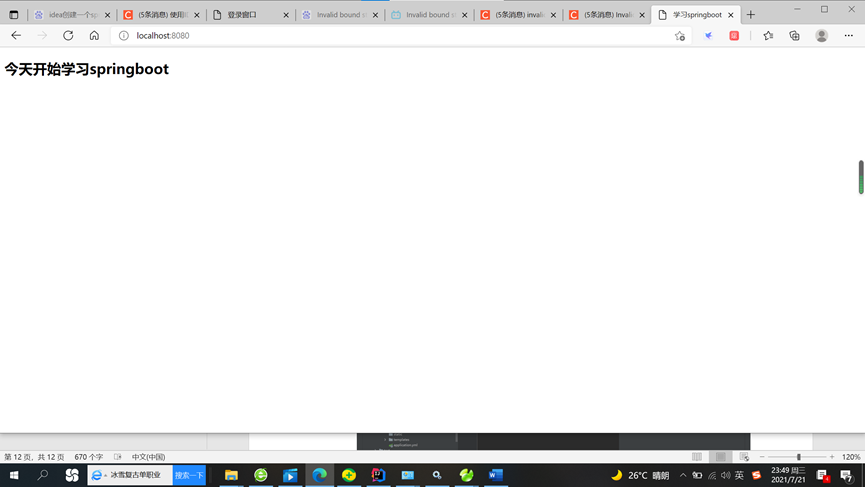




2021-07-21 09:14:27.473 WARN 9120---[main] o.m.s.mapper.ClassPathMapperScanner : No MyBatis mapper was found in '[com.example.test]' package. Please check your configuration.

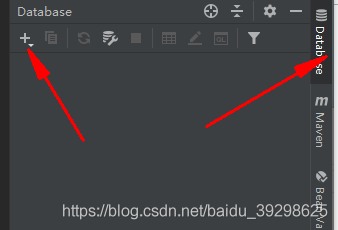
可以发现上面有一个WARN警告，那是因为还没有配置编写MyBatis的相关文件，下面会进行详解；

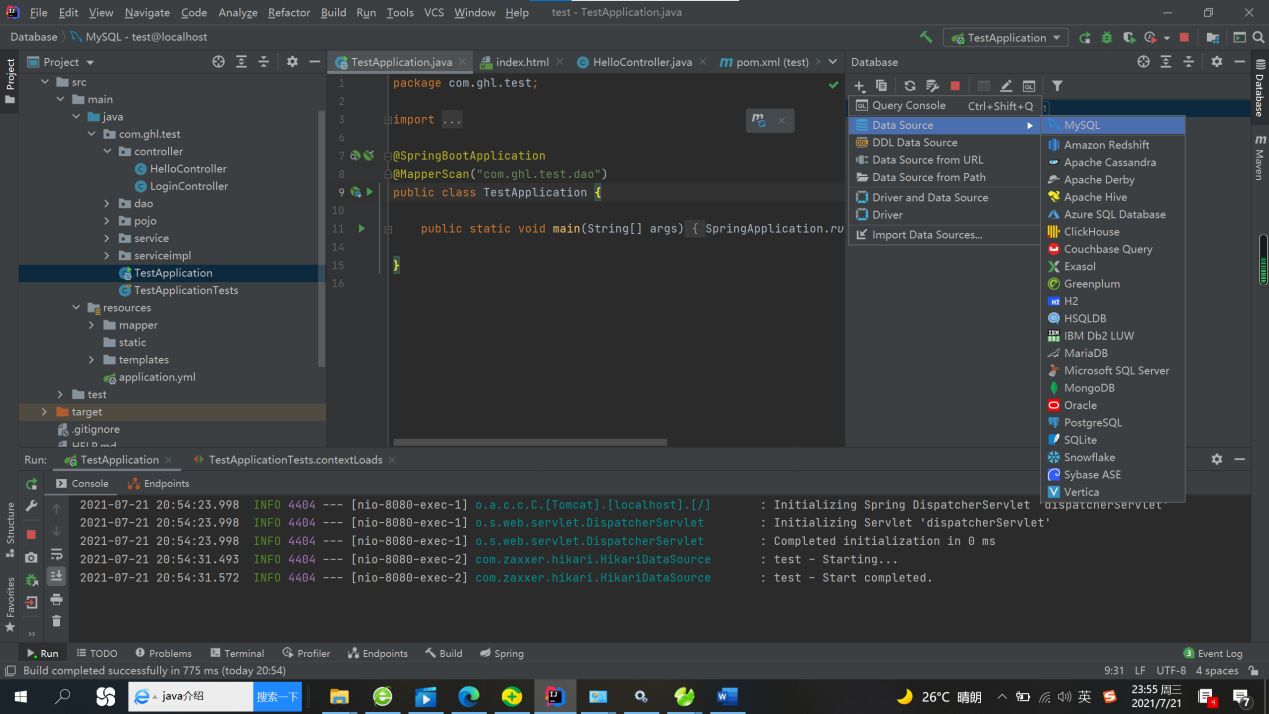
14.在浏览器中输入localhost:8080，回车显示初始的index界面；到这项目的初步搭建已经完成，下面可以下一些简单的业务逻辑，比如从数据库获取信息，登录之类的简单功能。



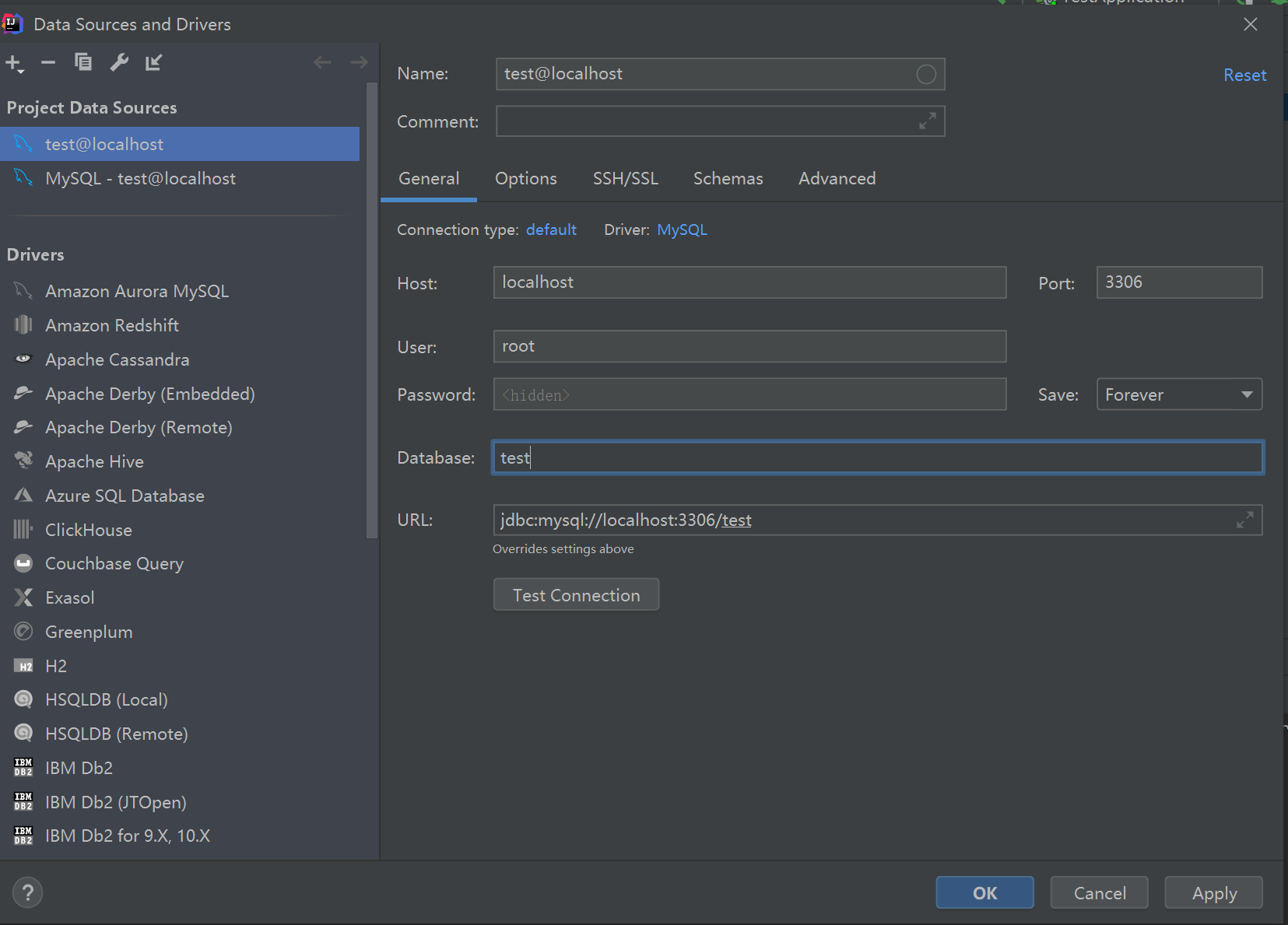
**六、数据库配置**

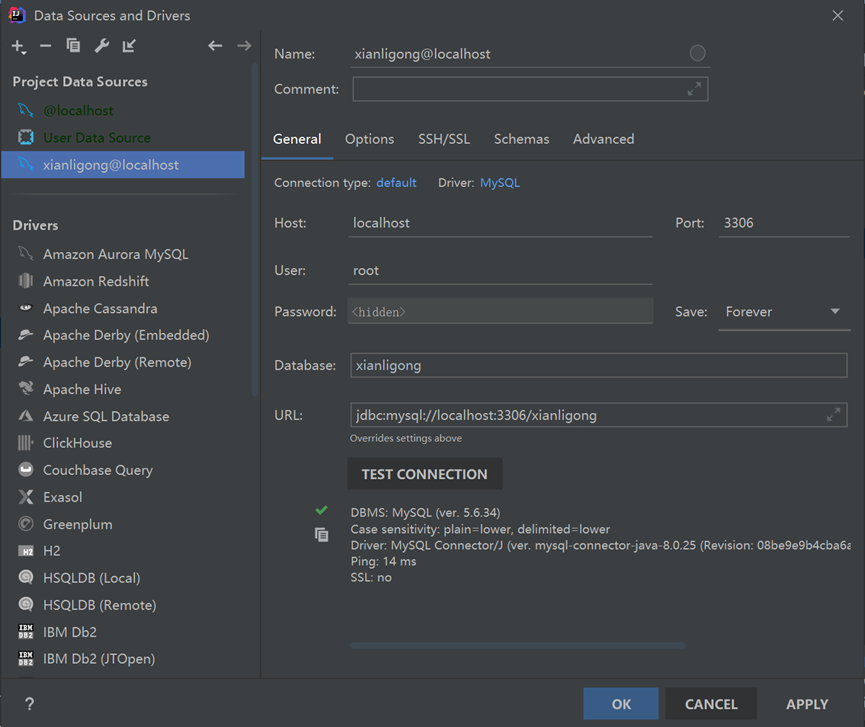
15.在进行下一步学习时，我们先来链接一下数据库；点击右侧的【Database】，点“+”，新建数据库链接；



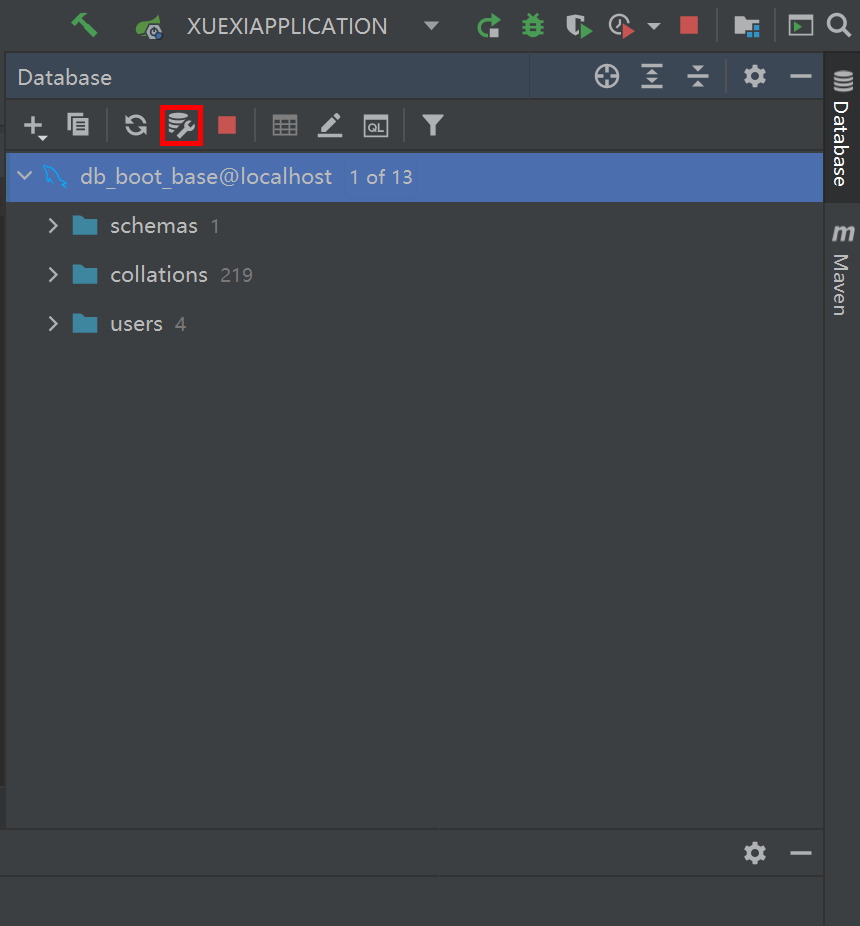


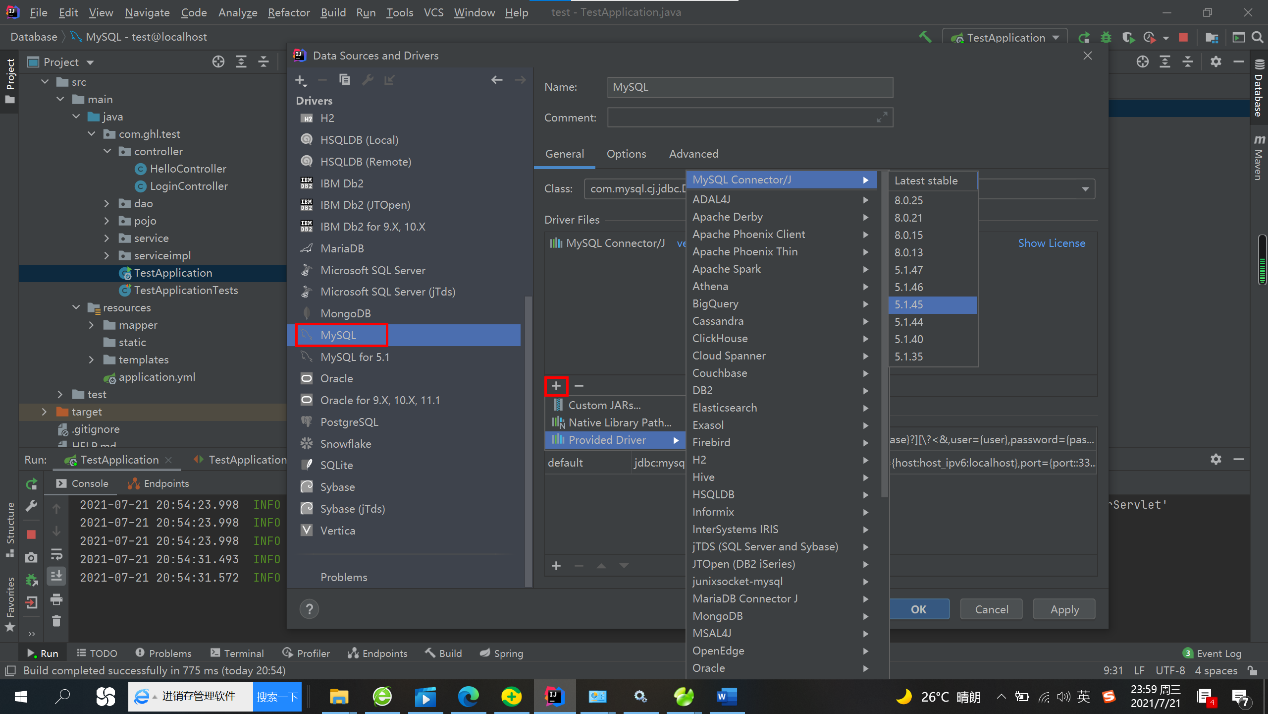
16.填写数据库相关信息，点击【Test Connection】；



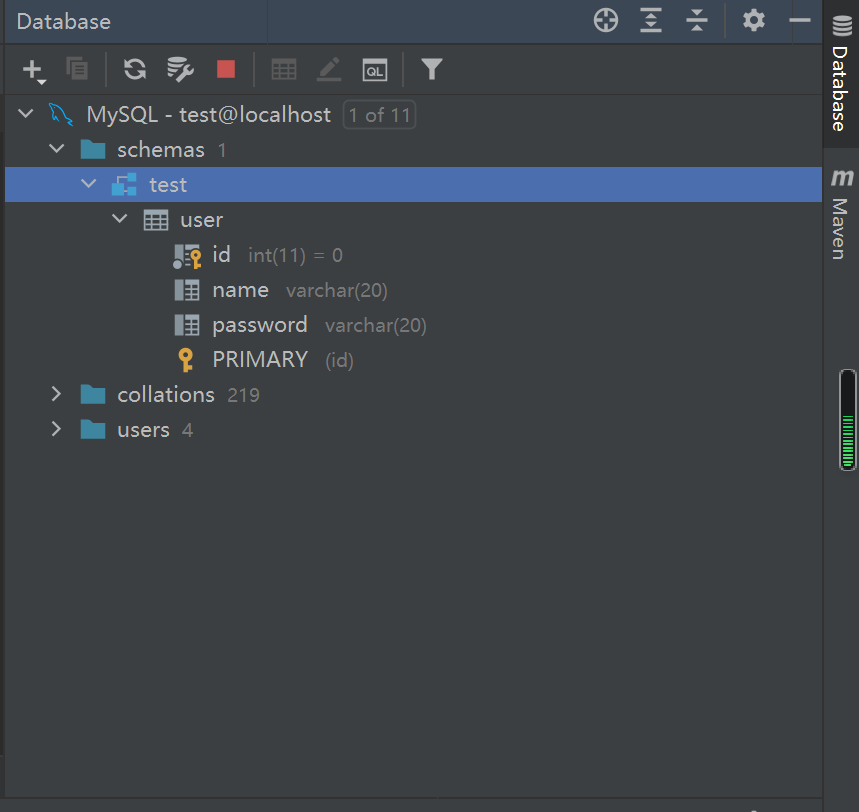


17.如果数据库连接失败，可能是驱动的问题，点击左上角的【+】，进入【Data Source and Drivers】界面，选择MySQL数据库，点击【+】，选择合适JDBC驱动程序，并下载。注意：选择合适版本驱动，记住版本号；





18.连接成功后，显示数据库信息，显示user表的基本信息；



**七、创建三层体系结构**

三层架构由用户界面层/表示层（User Interface），业务逻辑层（Business Logic Layer）和数据访问层（Data Acess Layer）构成。

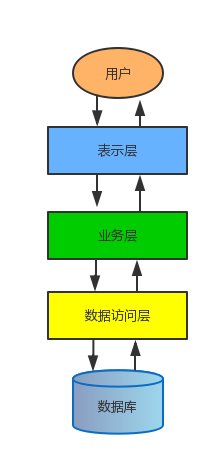
三层架构功能：

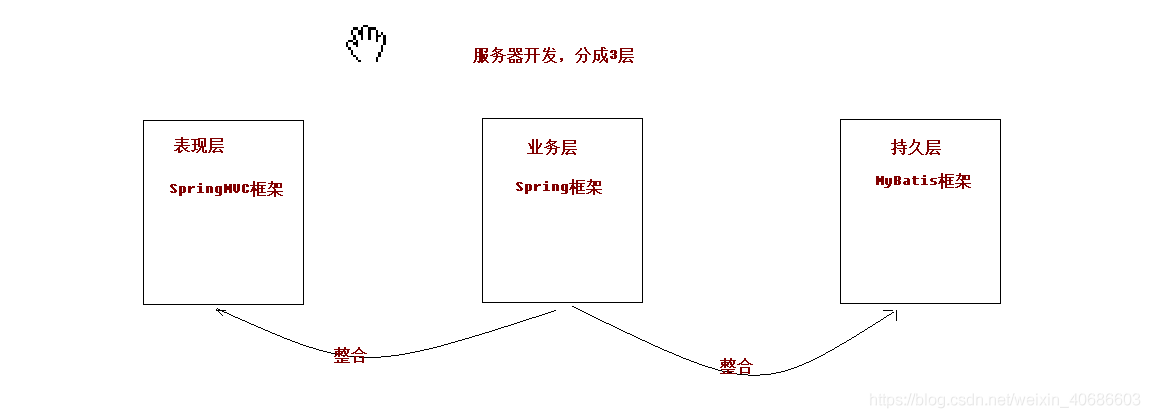
1. 用户界面层：负责与用户进行交互；
2. 业务逻辑层：主要是针对具体的问题的操作，也可以理解成对数据层的操作，对数据业务逻辑处理；
3. 数据访问层：主要是对非原始数据（数据库或者文本文件等存放数据的形式）的操作层，也就是说是对数据库，而不是对数据的操作，具体为业务逻辑层或用户界面层提供数据服务。

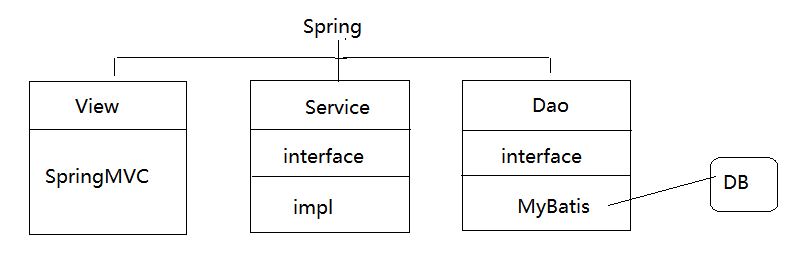
View层：表现层、表示层、controller层、用户界面层**、handler层。**

业务层：Service层、Server层、业务逻辑层

数据访问层：Dao层、持久层、Mapper层、Mapping层

****

****



19.SpringBoot项目大概分为四部分：

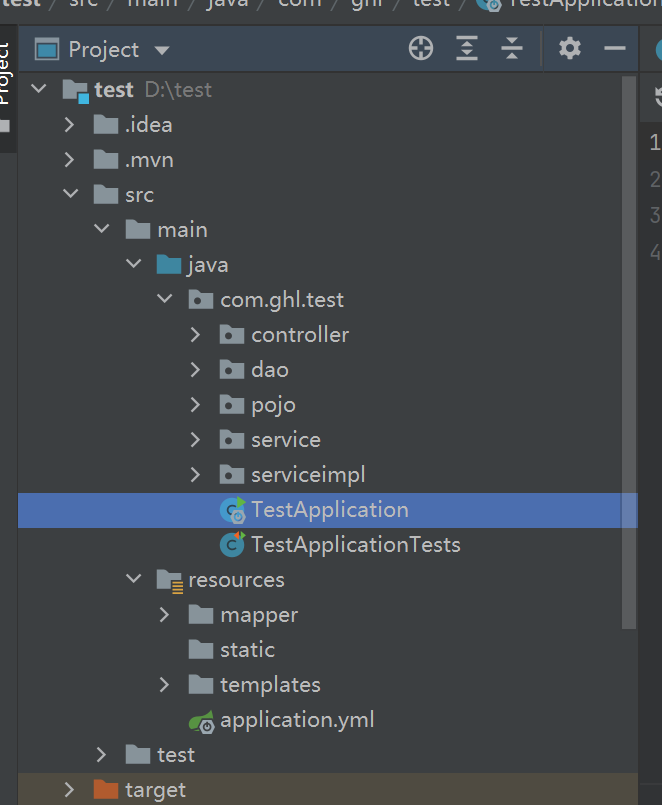
（1）Bean层：也叫model层，模型层，entity层，实体层，pojo层，bean层，ov层。指数据库表映射的实体类，存放JavaBean对象；本案例采用包pojo。

（2）DAO层：包括XxxMapper.java或XxxDao.java (数据库访问接口类)，XxxMapper.xml(数据库连接实现)；（这个命名，有人喜欢用Dao命名，有人喜欢用Mapper，看个人习惯），本案例采用XxxDao.java形式，即包是dao。

（3）Service层：也叫服务层，业务逻辑层，包括XxxService.java(业务接口类)，XxxServiceImpl.java（业务实现类）；（可以在service包下新建impl文件存放业务实现类，也可以把业务实现类单独放一个包下，更清晰）。本案例采用service和serviceImpl两个包分别存放形式。

（4）Web层：就是Controller层，实现与web前端的交互。本案例采用包controller。

依照上面四层，创建目录结构如下：



**7.1 MyBatis配置**

20.代码展示：

（1）在application配置文件中添加MyBatis配置：

spring:  
 datasource:  
 name: test *#数据库名* url: jdbc:mysql://localhost:3306/test?useUnicode=true&characterEncoding=UTF-8

*#url* username: root *#用户名* password: *#密码* driver-class-name: com.mysql.cj.jdbc.Driver *#数据库链接驱动*mybatis:  
 mapper-locations: classpath:mapper/\*.xml *#配置映射文件* type-aliases-package: com.ghl.test.pojo *#配置实体类别名*

**7.2 pom.xml文件配置**

（2）pom.xml文件配置信息（备注：这个文件以前没有，这个里面也添加了单元测试所需的配置，记得要重新导一下Maven包）

<?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>2.5.2</version>

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

</parent>

<groupId>com.ghl</groupId>

<artifactId>test</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>test</name>

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

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<dependency>

<groupId>org.mybatis.spring.boot</groupId>

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

<version>2.2.0</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.25</version>

<scope>runtime</scope>

</dependency>

<dependency>

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

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

<scope>test</scope>

</dependency>

<dependency>

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

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

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-test</artifactId>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>

**7.3 JavaBean实体类**

(3)Bean实体类，依据数据库表，生成setter和getter方法；

package com.ghl.test.pojo;

public class User {

private int id;

private String name;

private String password;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}

**7.4 DAO层（持久层）**

（4）DAO层访问数据库接口文件：

package com.ghl.test.dao;

import com.ghl.test.pojo.User;

public interface UserDao {

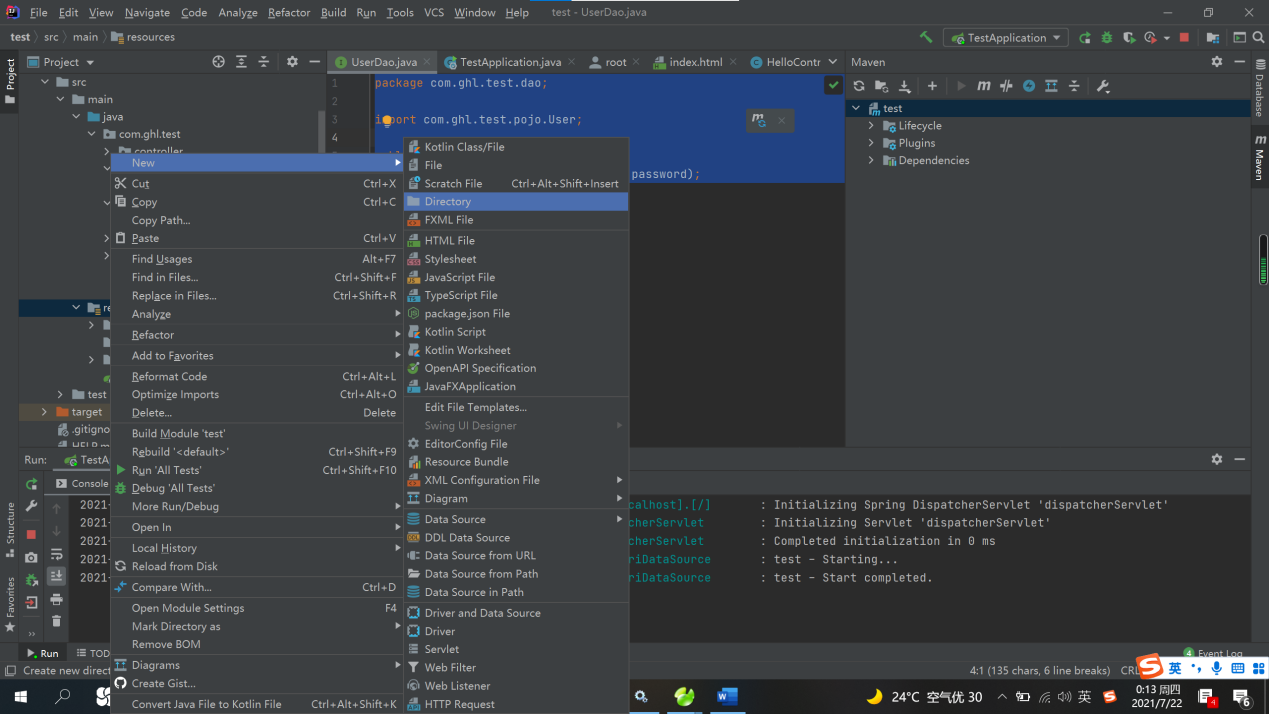
User getInfo(String name, String password);

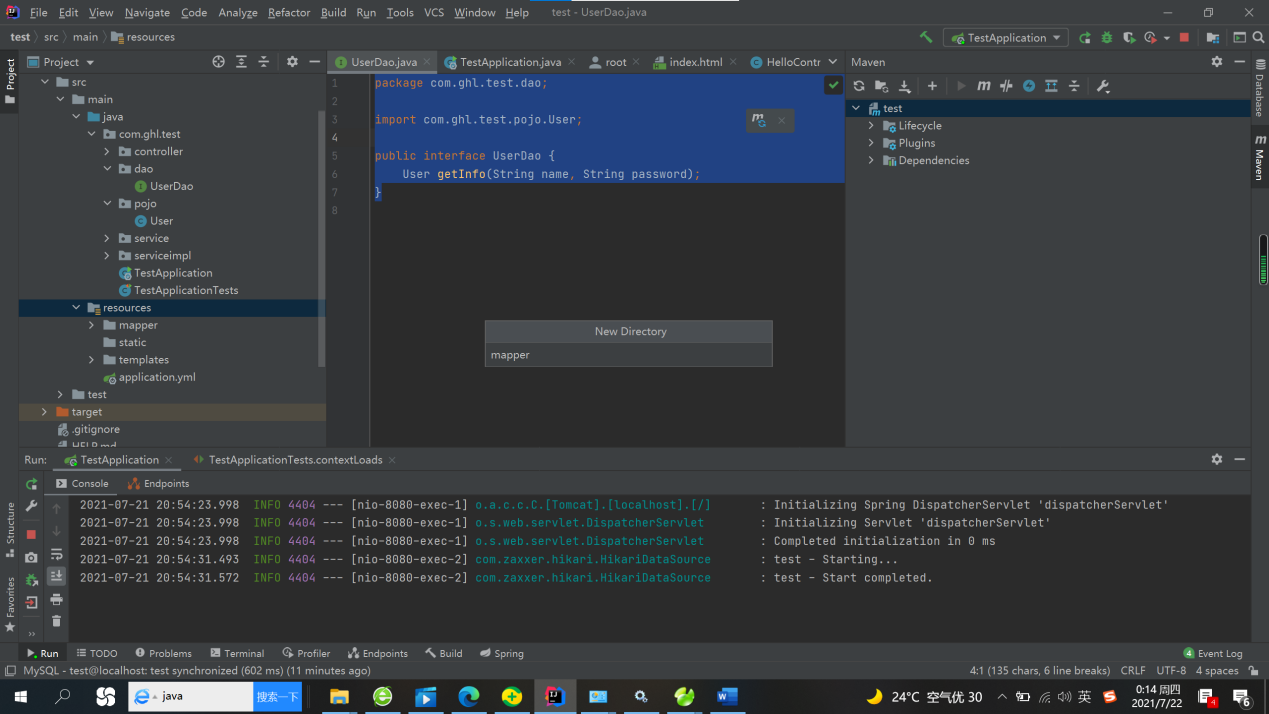
}

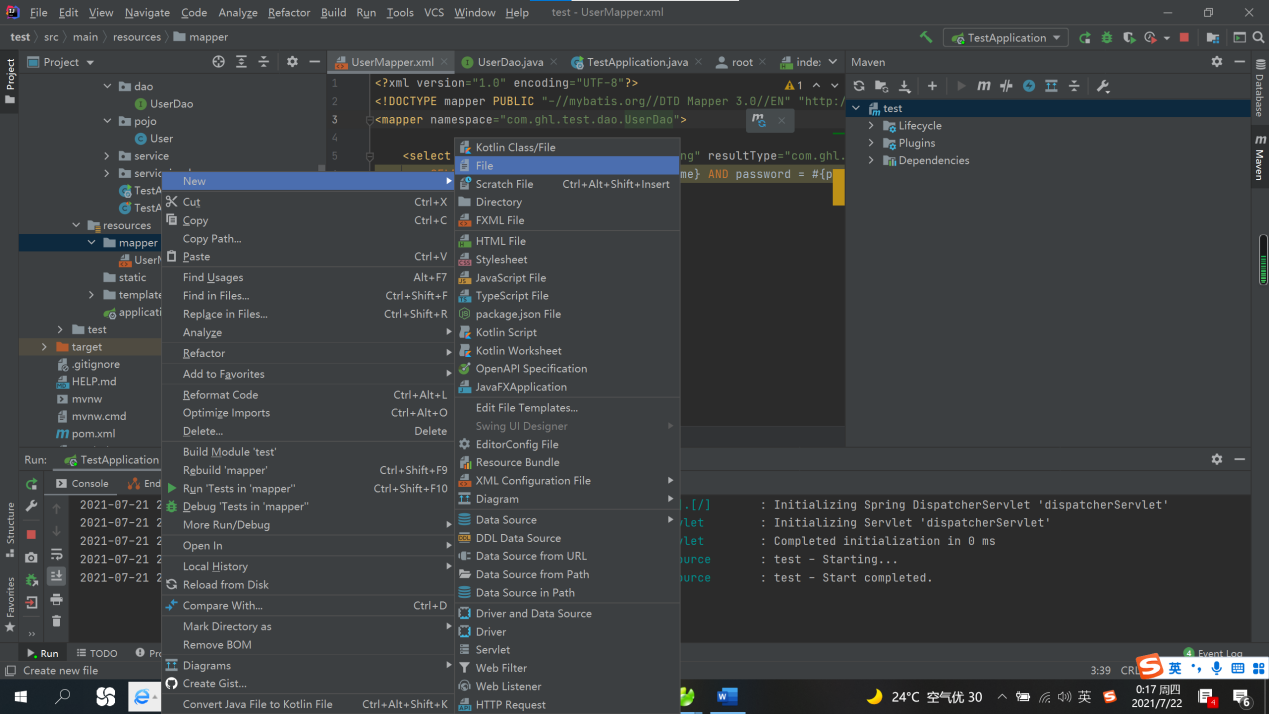
**7.5 Mapper映射文件**

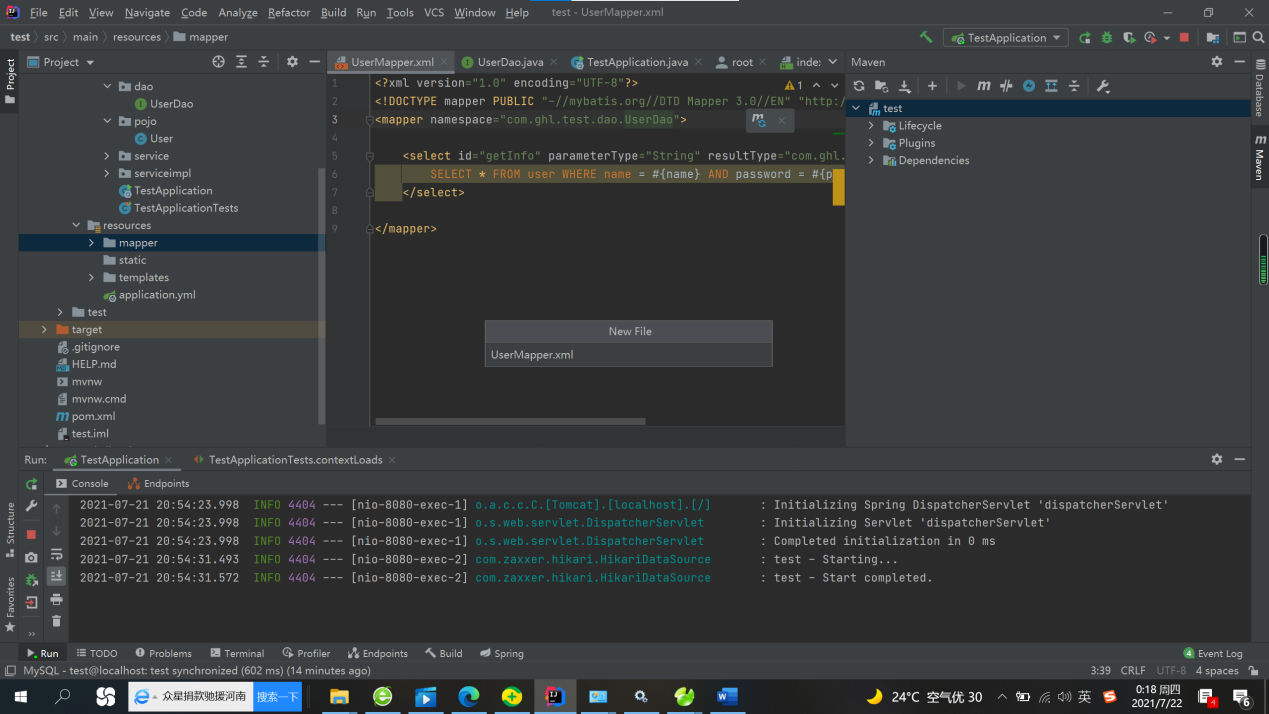
（5）DAO层访问数据库实现文件（需在resource包（类路径）下创建mapper目录，然后再创建一个UserMapper.xml。在application配置文件中mybatis:mapper-locations:对应的就是该文件地址）。可以把Mapper文件理解为持久层接口实现类。

注意：<mapper>标签的namespace属性要填写访问持久层接口类文件完全路径。









**UserMapper.xml代码如下：**

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

<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd" >

<mapper namespace="com.ghl.test.dao.UserDao">

<select id="getInfo" parameterType="String" resultType="com.ghl.test.pojo.User">

SELECT \* FROM user WHERE name = #{name} AND password = #{password}

</select>

</mapper>

注意：

namespace属性是对应的持久层接口的完全类名，即“包名+类名”。

id必须与持久层（Dao）接口中对应方法体名称完全保持一致，包括大小写。

application.yml代码如下：

spring:

datasource:

name: test #数据库名

url: jdbc:mysql://localhost:3306/test #url

username: root #用户名

password: #密码

driver-class-name: com.mysql.cj.jdbc.Driver #数据库链接驱动

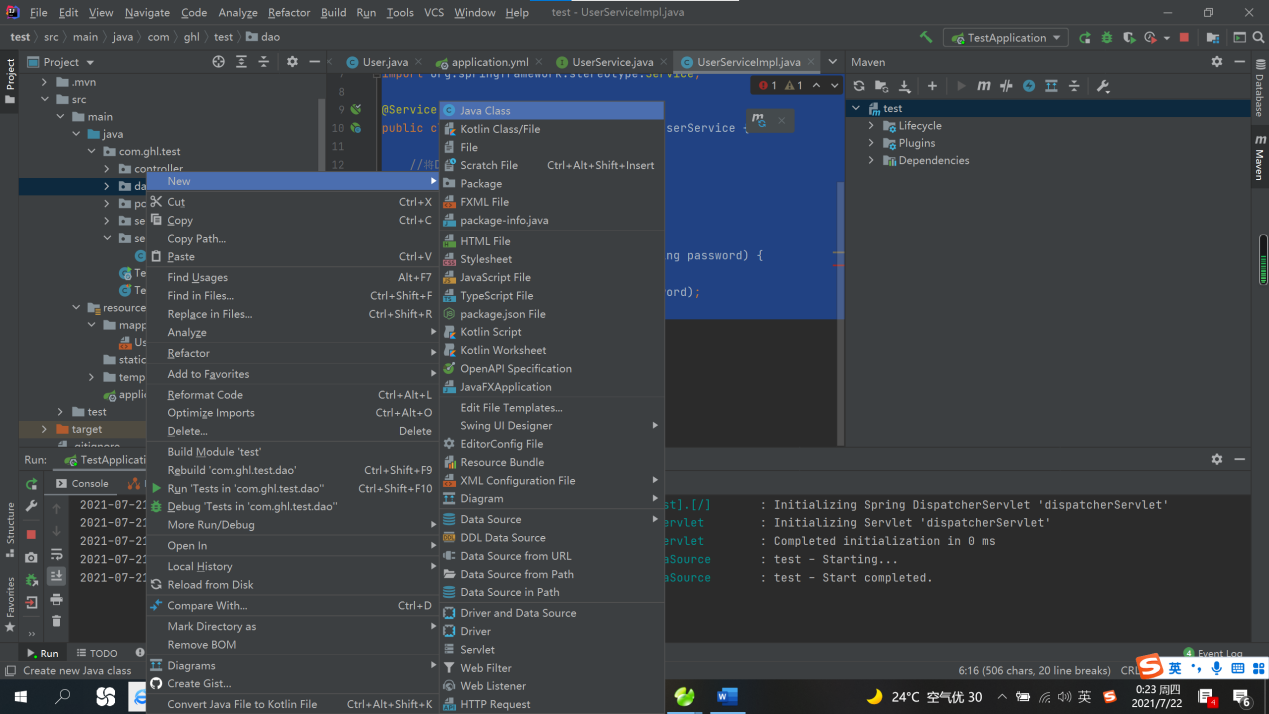
mybatis:

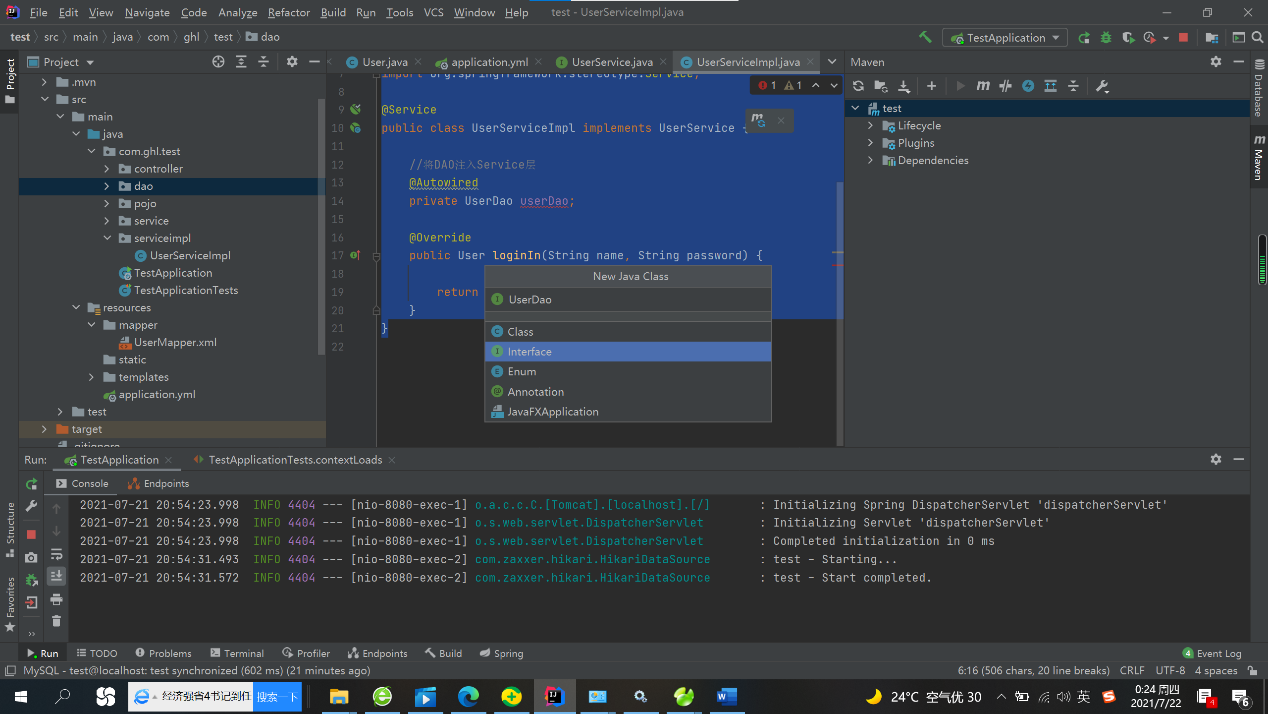
mapper-locations: classpath:mapper/\*.xml #配置映射文件

type-aliases-package: com.ghl.test.pojo #配置实体类别名

**7.6 Service层（业务逻辑层）**

（6）Service层业务接口类编写：





package com.ghl.test.service;

import com.ghl.test.pojo.User;

public interface UserService {

User loginIn(String name, String password);

}

（7）Service层业务实现类编写，注意要注解@Service，注入DAO：

package com.ghl.test.serviceimpl;

import com.ghl.test.dao.UserDao;

import com.ghl.test.pojo.User;

import com.ghl.test.service.UserService;

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

import org.springframework.stereotype.Service;

@Service

public class UserServiceImpl implements UserService {

//将DAO注入Service层

@Autowired

private UserDao userDao;

@Override

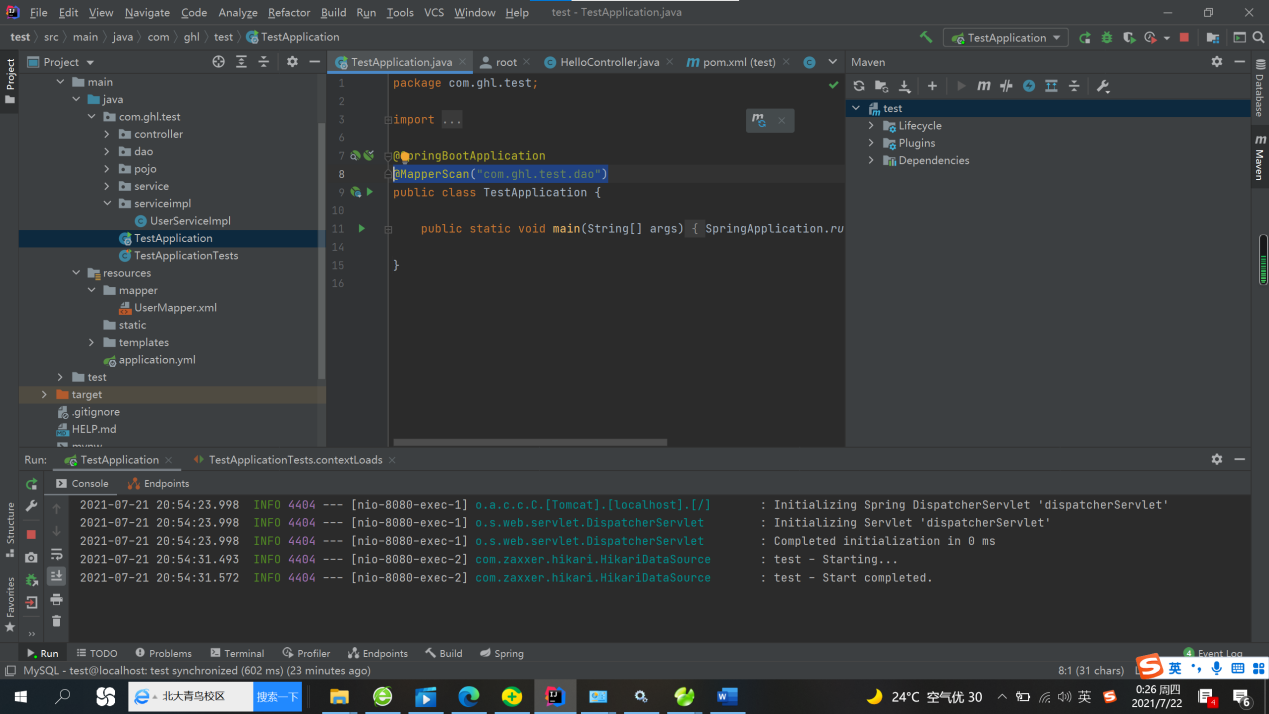
public User loginIn(String name, String password) {

return userDao.getInfo(name,password);

}

}

（8）项目启动类要添加注解@MapperScan，项目启动时扫描Dao（mapper）接口，否则会报错找不到Dao（mapper）文件：



代码如下：

package com.ghl.test;

import org.mybatis.spring.annotation.MapperScan;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

@MapperScan("com.ghl.test.dao")

public class TestApplication {

public static void main(String[] args) {

SpringApplication.run(TestApplication.class, args);

}

}

@SpringBootApplication注解是SpringBoot框架的核心注解，该注解用于表明是SpringBoot项目的主程序启动类。SpringApplication.run(TestApplication.class, args)；调用SpringApplication.run()方法启动主程序类。

**7.7 Controller层（表示层）**

（9）controller层，注意添加@controller注解，注入Service服务：

**LoginController.java代码如下：**

package com.ghl.test.controller;

import com.ghl.test.pojo.User;

import com.ghl.test.service.UserService;

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

import org.springframework.stereotype.Controller;

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

import org.springframework.web.bind.annotation.RequestMethod;

@Controller

public class LoginController {

//将Service注入Web层

@Autowired

UserService userService;

@RequestMapping("/login")

public String show(){

return "login";

}

@RequestMapping(value = "/loginIn",method = RequestMethod.POST)

public String login(String name,String password){

User user = userService.loginIn(name,password);

if(user!=null){

return "success";

}else {

return "error";

}

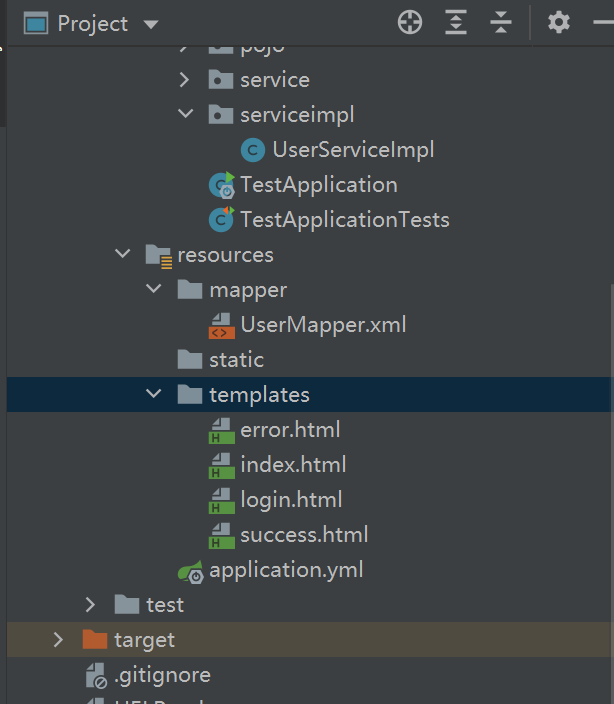
}

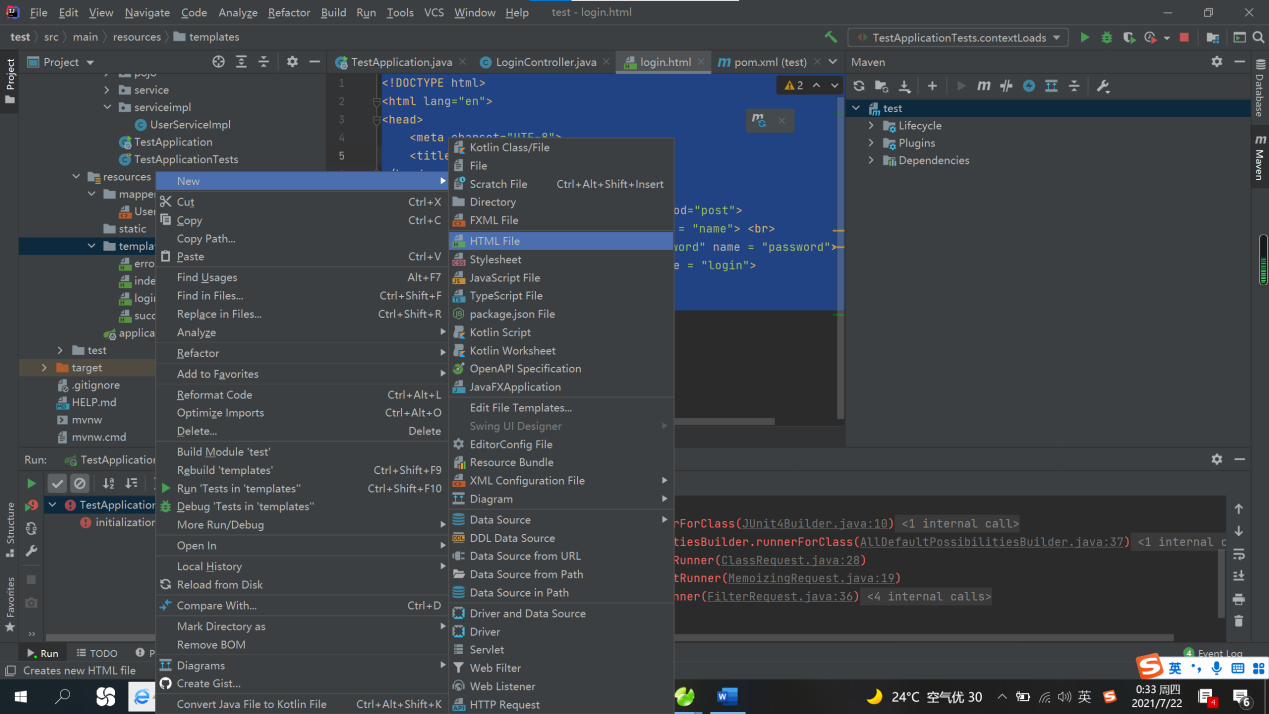
}

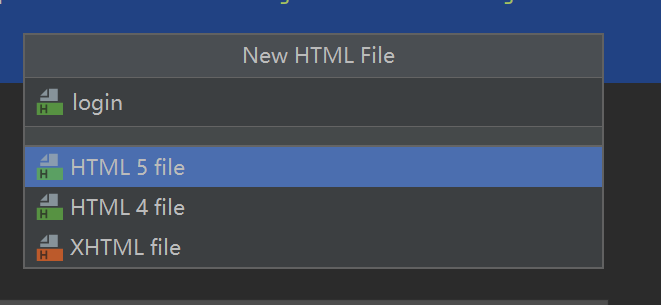
**7.8 登录界面**

（10）html文件：

该案例所有html文件均放在templates目录下：



****

****

**login.html代码如下：**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>登录窗口</title>

<style>

#myDiv{

width: 400px;

height: 400px;

position: absolute;

left: 50%;

top: 50%;

margin: -200px 0 0 -200px;

}

</style>

</head>

<body>

<div id="myDiv">

<fieldset width="400">

<legend>登录窗口</legend>

<form role="form" action = "/loginIn" method="post">

账号：<input type="text" id="name" name = "name"> <br>

密码：<input type="password" id = "password" name = "password"> <br>

<input type="submit" id = "login" value = "login">

</form>

</fieldset>  
</div>

</body>

</html>

**success.html代码如下：**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>成功窗口</title>

</head>

<body>

<h1>登录成功！</h1>

</body>

</html>

**error.html代码如下：**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>失败窗口</title>

</head>

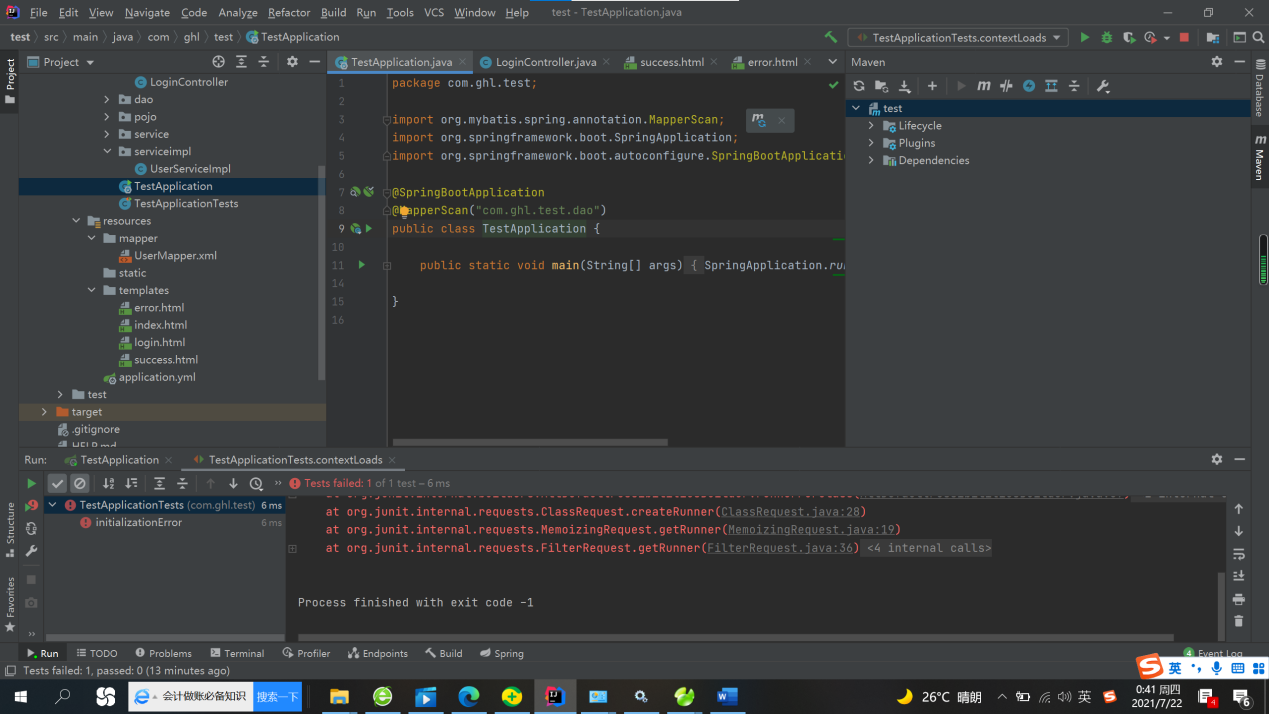
<body>

<h1>登录失败！</h1>

</body>

</html>

23.运行TestApplication.java文件，启动项目



**7.9 MySQL数据库驱动异常处理**

24.发现一条警告信息，是数据库连接的jar包问题：

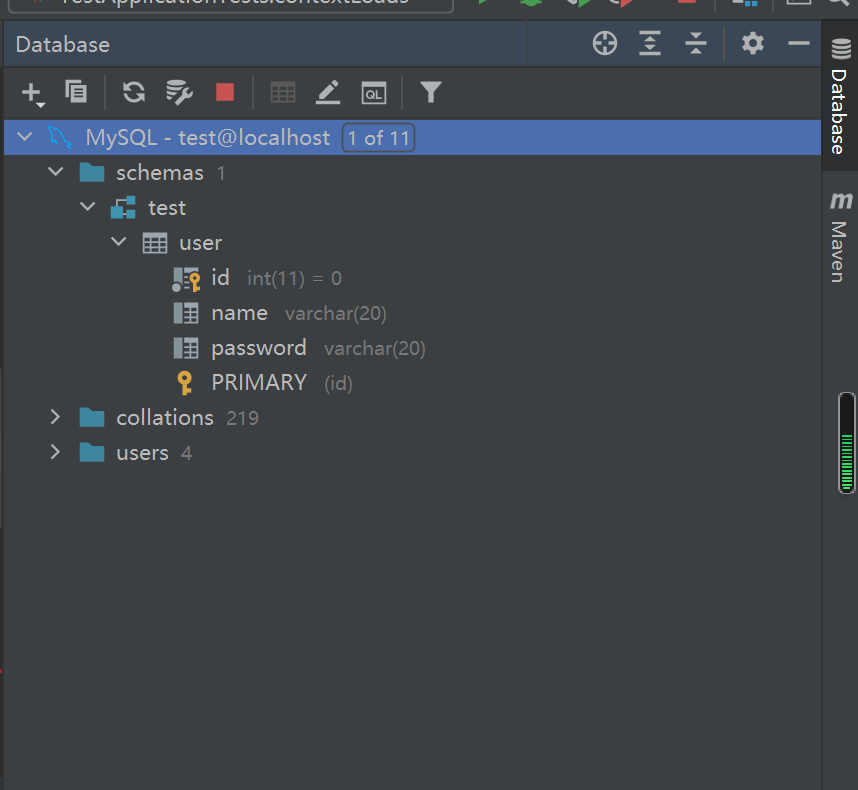
2019-08-02 11:25:04.150 WARN 16868 --- [ main] com.zaxxer.hikari.util.DriverDataSource : Registered driver with driverClassName=com.mysql.jdbc.Driver was not found, trying direct instantiation.

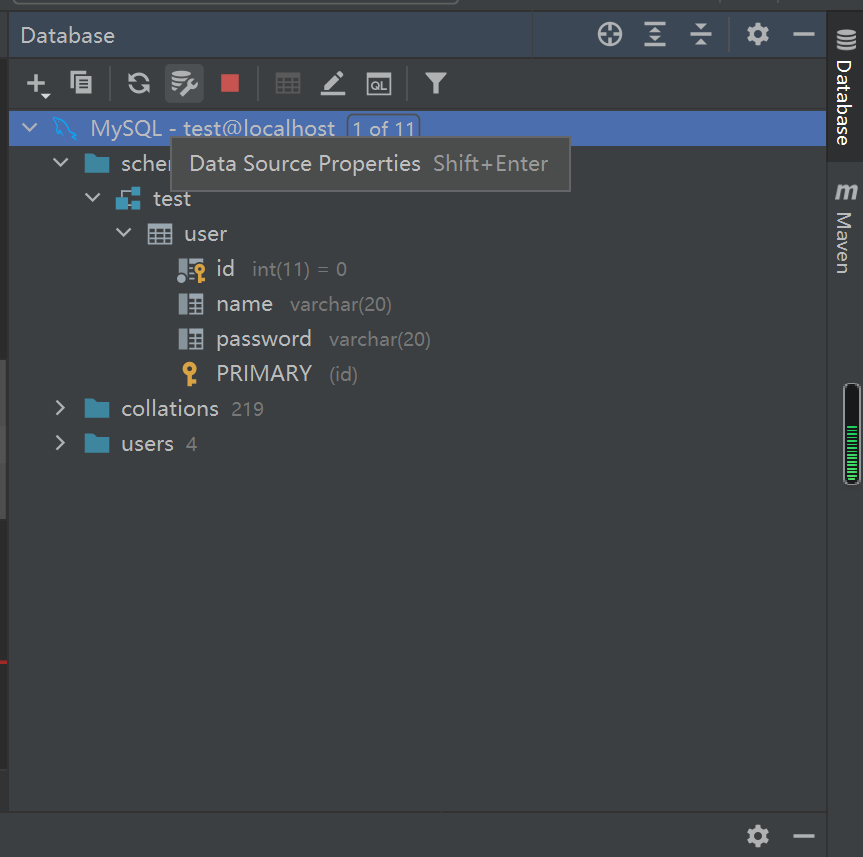
打开pom.xml文件，发现配置文件中未指定数据库连接的jar包的版本号，用version标签引入。

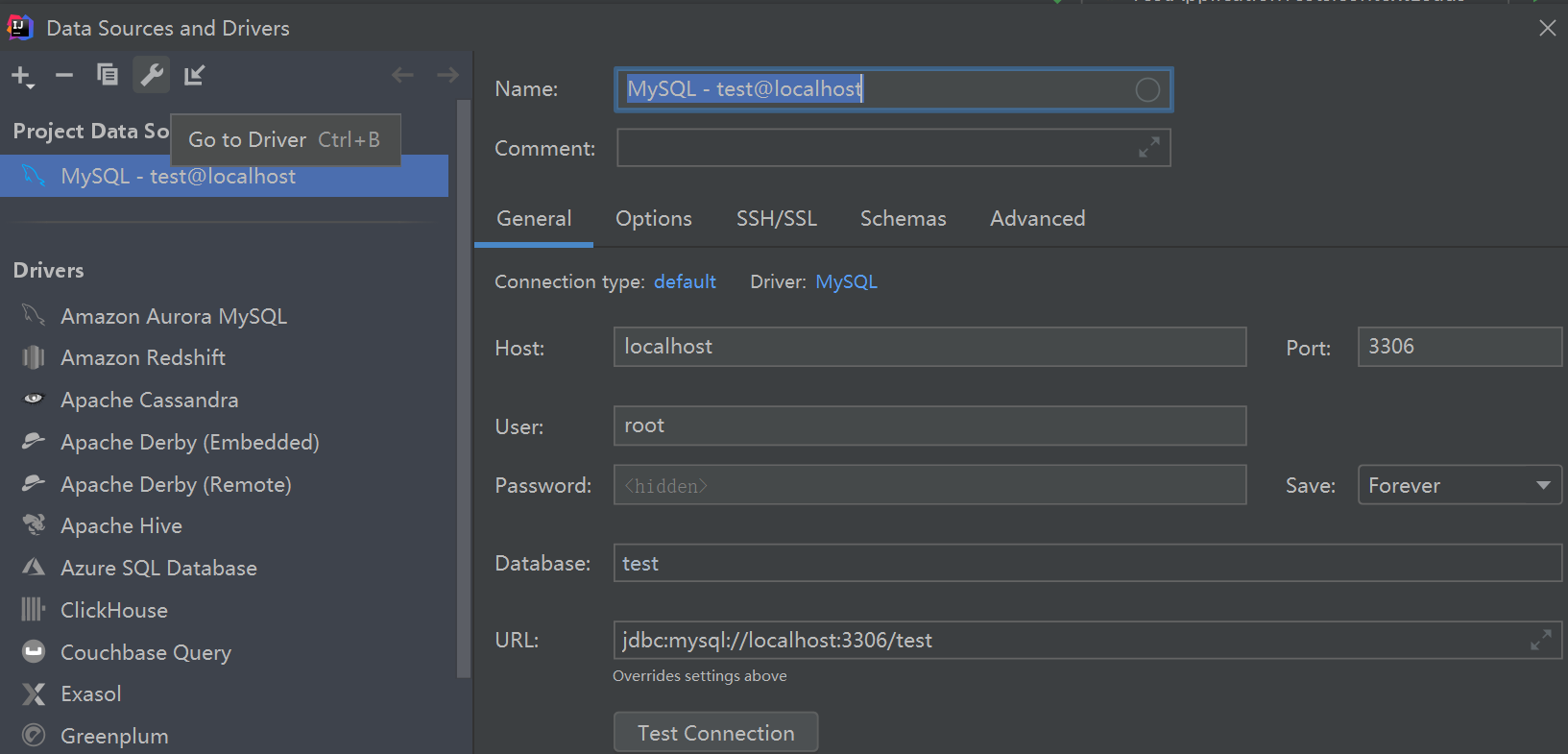


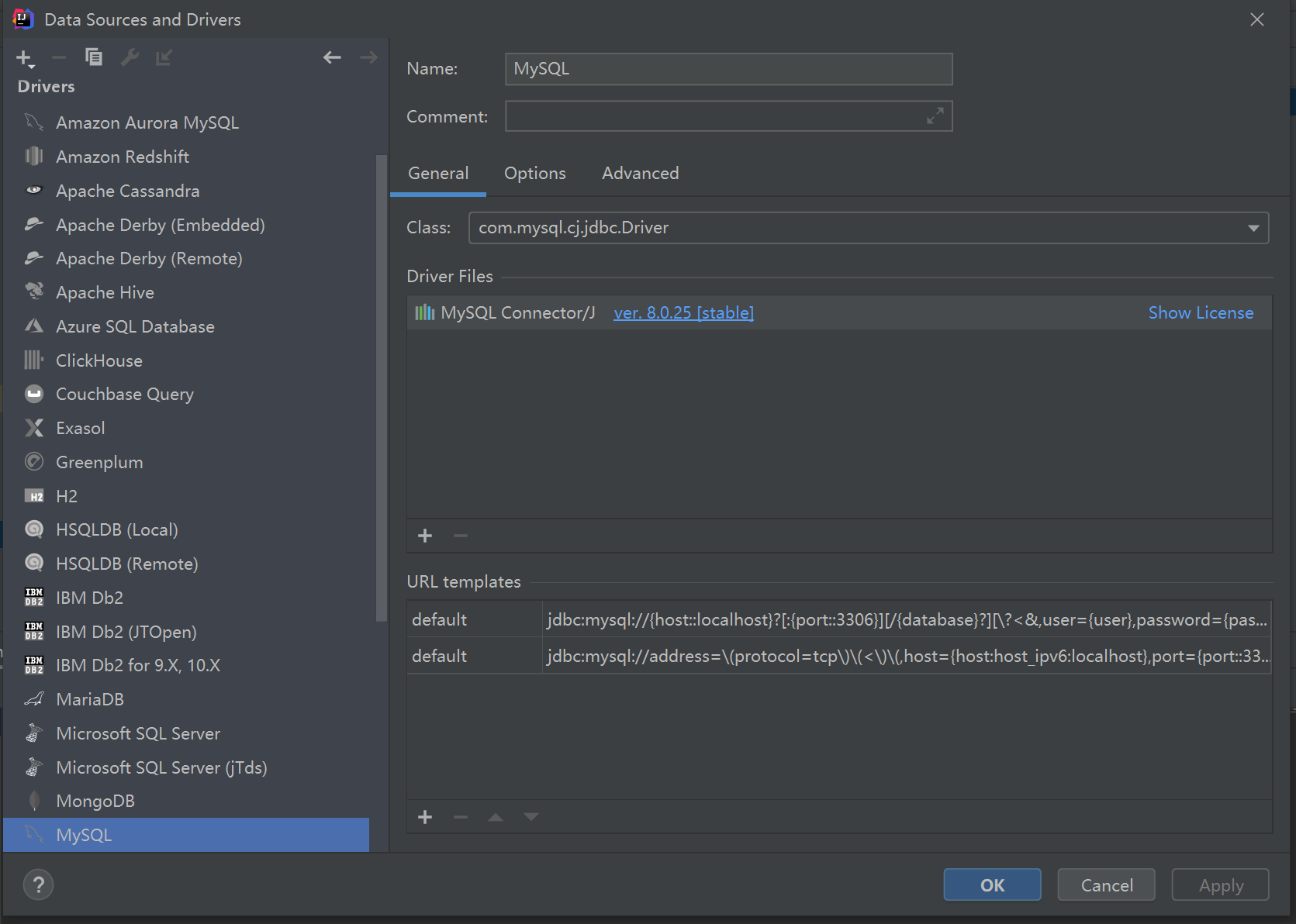
<version>8.0.25</version>

具体查询MySQL数据库jdbc驱动版本号步骤如下：

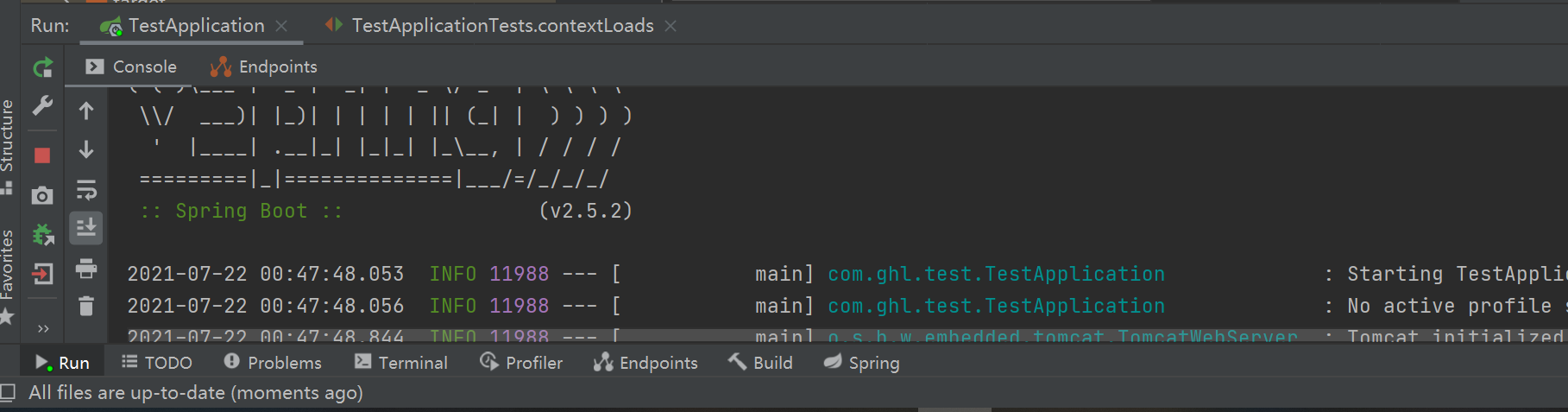






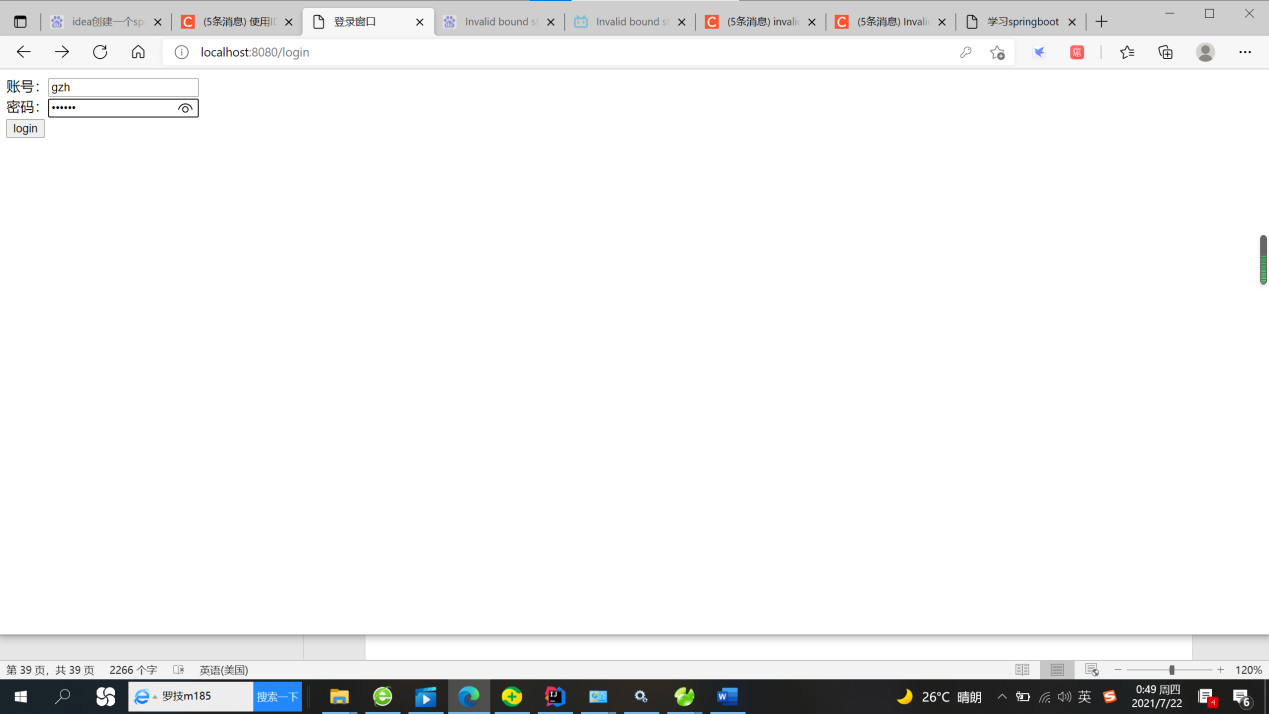


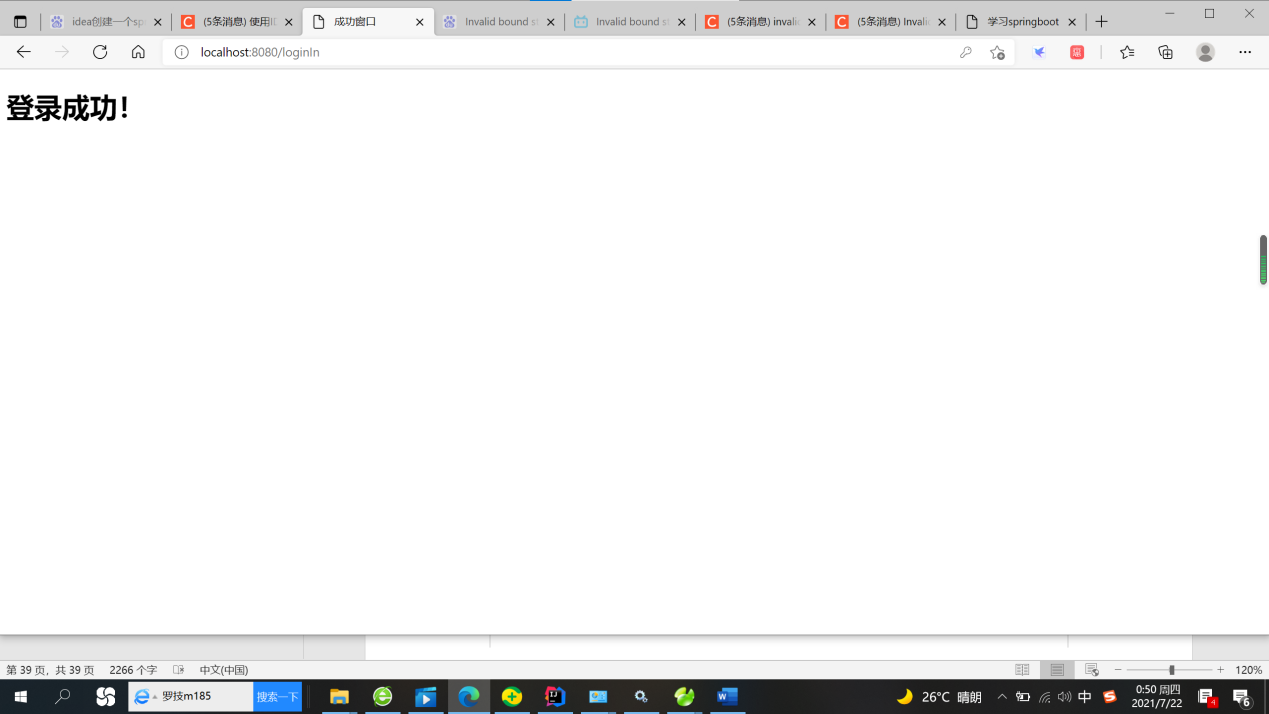
25.运行TestApplication.java文件，启动项目，无任何WARN警告信息，进入浏览器输入localhost:8080/login

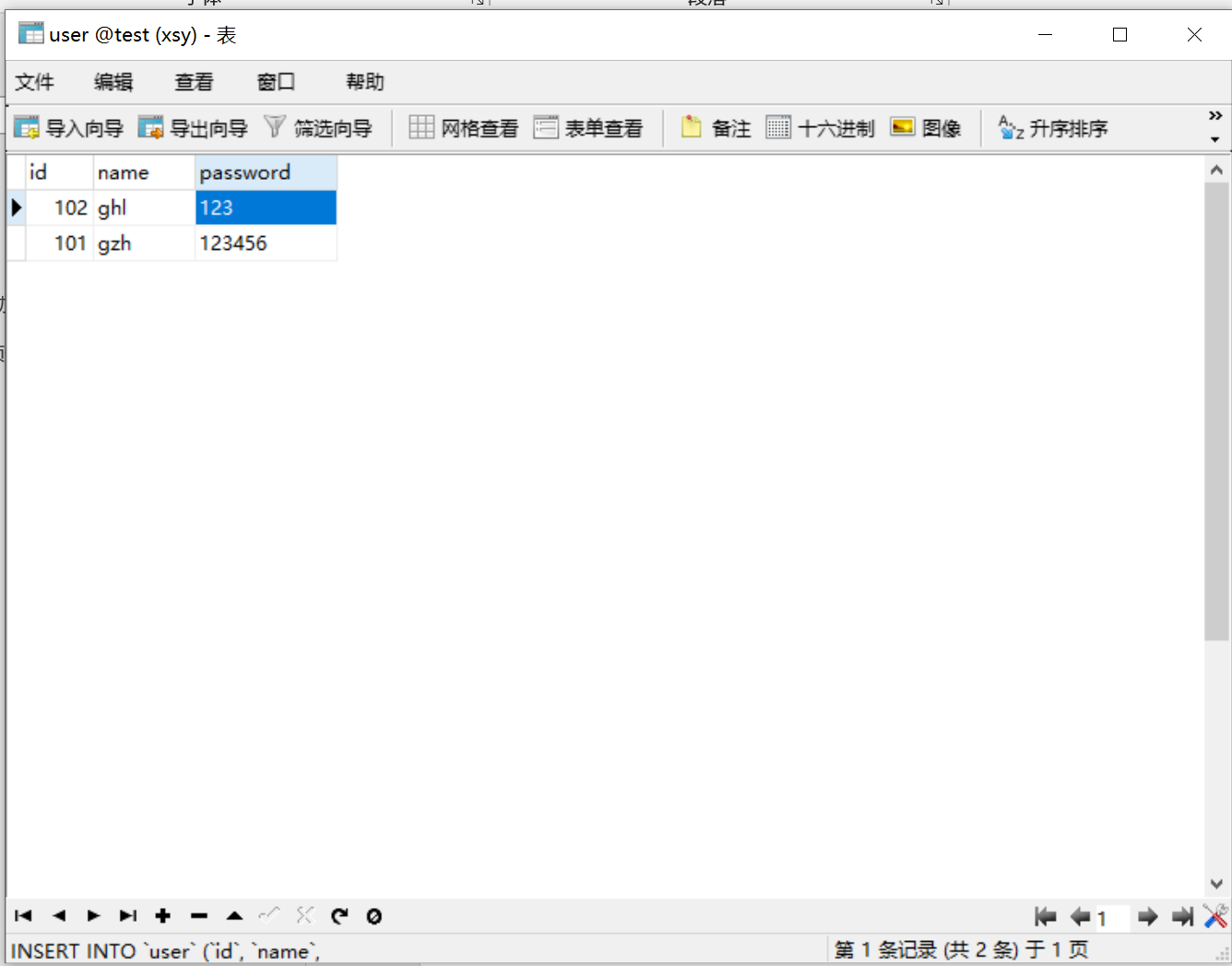




输入账号：gzh 密码：123456，正确用户名和密码。







输入账号：gzh 密码：123123，用户名或密码错误。

