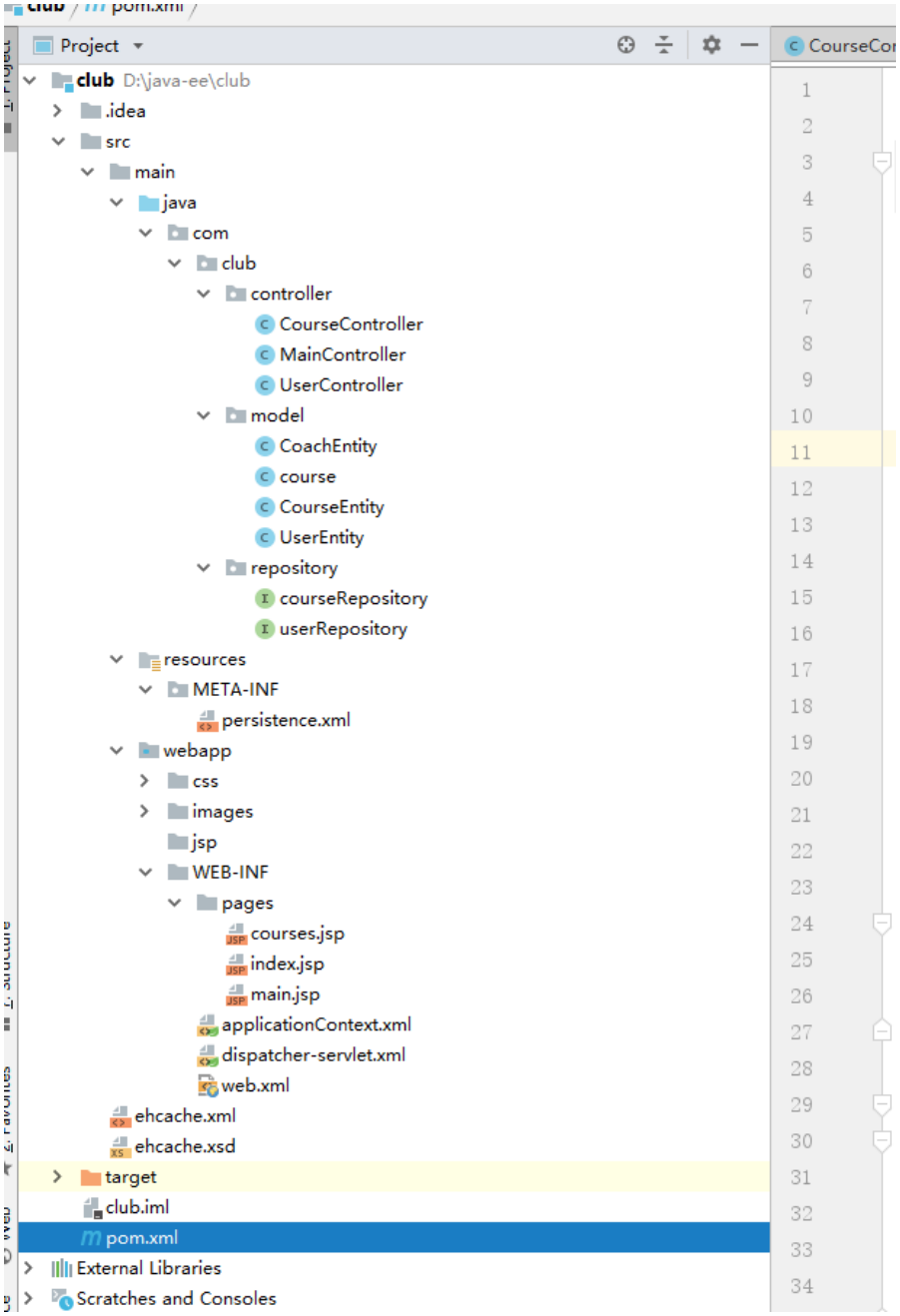
1. 项目描述

SpringMvc+Maven+Spring Data Jpa(由 Hibernate JPA 提供)+Cache 集合而成的项目。

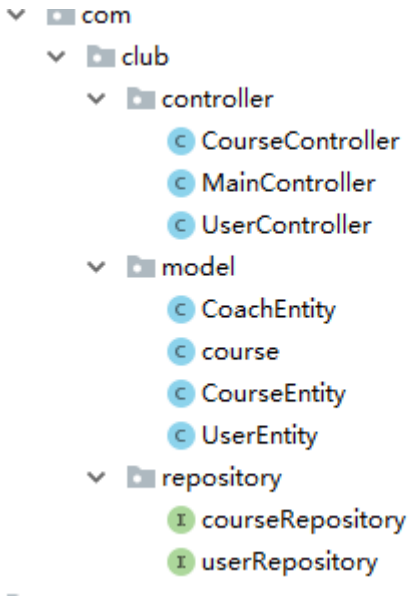
项目结构图如下：



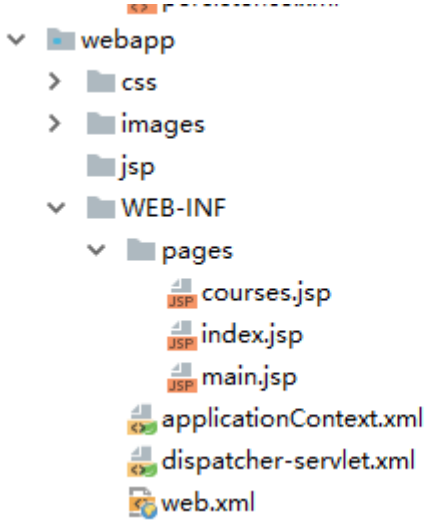
2. 要点分析

（1）SpringMvc 框架的使用

三层结构如图所示：



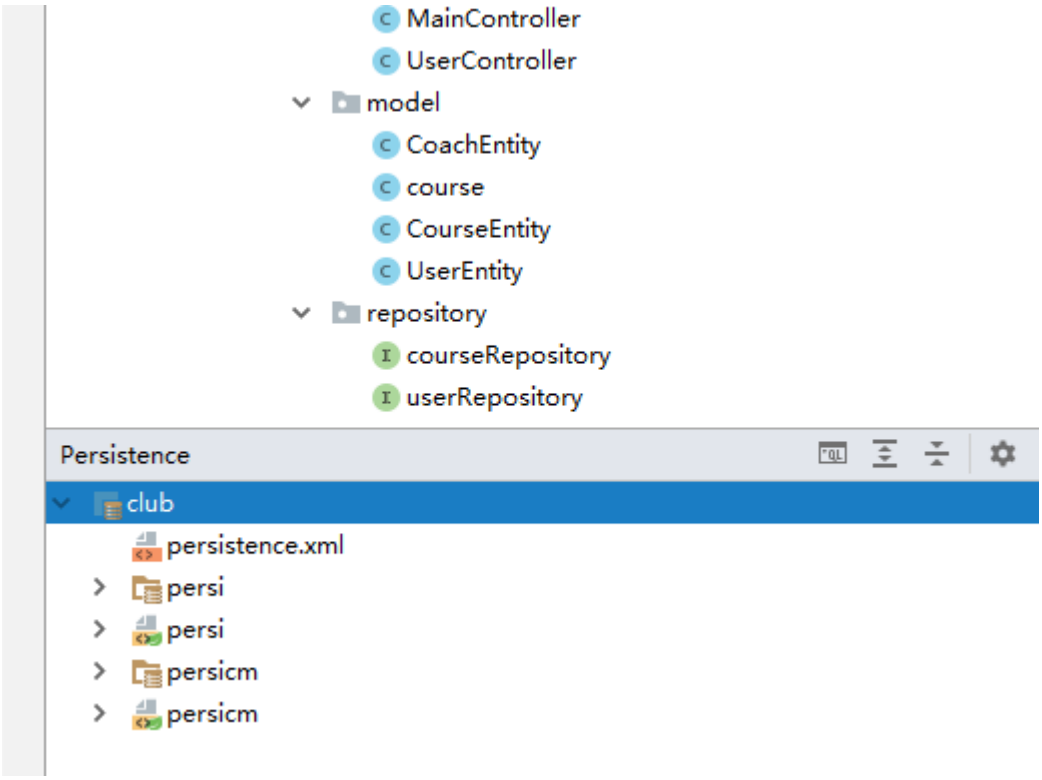
资源文件如图所示：



页面跳转基本在 Controller 中实现：

```
    ModelAndView mav = new ModelAndView( viewName: "main");
    mav.addObject( attributeName: "username",username);
    return mav;
}
return new ModelAndView( viewName: "index");
```

（2）Spring Data Jpa(由 Hibernate JPA 提供)的使用



在项目的中的具体体现：

简单获取：

```
List<UserEntity> list=new ArrayList<~>();
list=(List<UserEntity>) userRepository.findAll();
```

多表使用：

在 Repository 接口中自定义多表查询，以查询一个学员全部其教练的等级高于 2 的所有课程（在页面中为高级课程）

```
ıblic interface courseRepository extends JpaRepository<CourseEntity, Integer> {
  @Transactional
  // 定义查询
  // @Param注解用于提取参数
  @Query("select c from CourseEntity c, CoachEntity h where c.coachName=:qName and h.coachName=:qName and h.coachLevel>2")
  public List<CourseEntity> getHigh(@Param("qName") String name);
```

（3）缓存（基于 SpringMVC 实现）

使用@Cacheable 注解实现注释：

```
@Cacheable(value="cachedlist")
public List<CourseEntity> cache_course(String loged) {
    List<CourseEntity> list1=new ArrayList<~>();
    list1=(List<CourseEntity>) courseRepository.findAll();
    for(CourseEntity test:list1) {
        if(!test.getUserName().equals(loged)) {
            list1.remove(test.getCourseId());
        }
    }
    return list1;
}
```

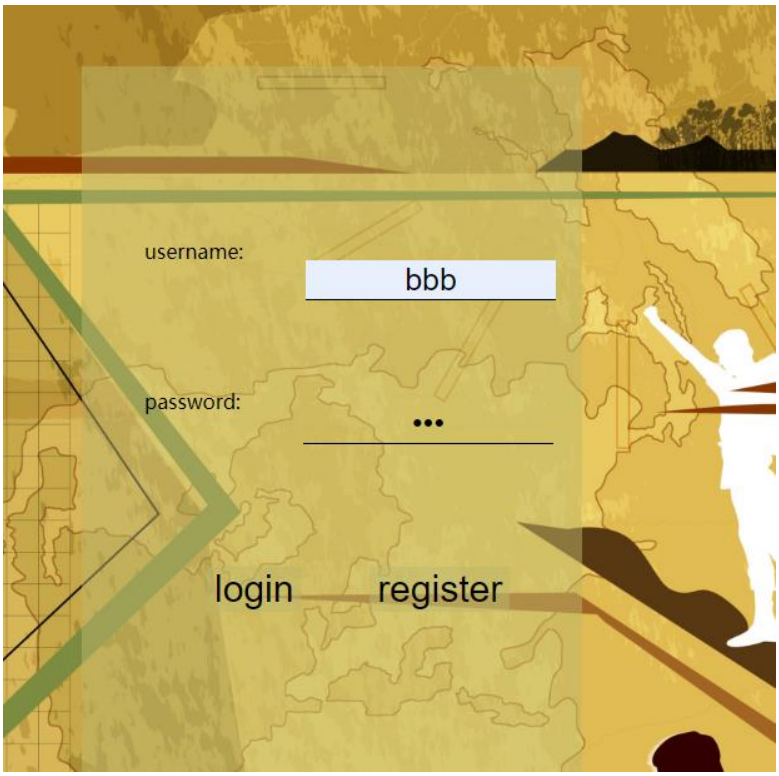
此注释的调用：

红框中即为从项目缓存中获取以该学员的 username 为 key 的缓存的值，结果即为其全部（高级）课程，用于在页面中直接显示而不需要再次访问数据库，则加快了响应速度

```
    System.out.println("多表："+courseEntity.getCourseId()+" "+courseEntity.get(
}
ModelAndView mav = new ModelAndView( viewName: "courses");
mav.addObject( attributeName: "courselist",cache_course(loged));
return mav;
```

（4）功能展示

【1】登录



【2】场馆展示

【3】课程展示

1	tennis	2h	bbb	lean	修改 | 查看
132	tennis	2h	bbb	lean	修改 | 查看