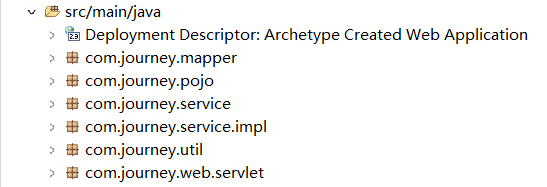
**后端开发文档**

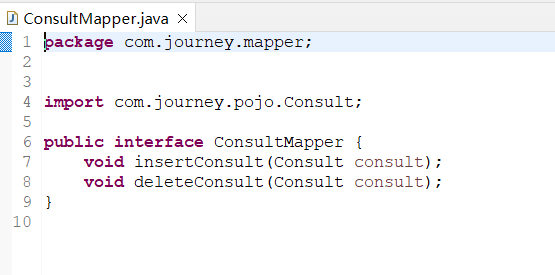
后端开发部分主要遵循MVC模式，即Model（模型）、View(视图)、Controller（控制器）



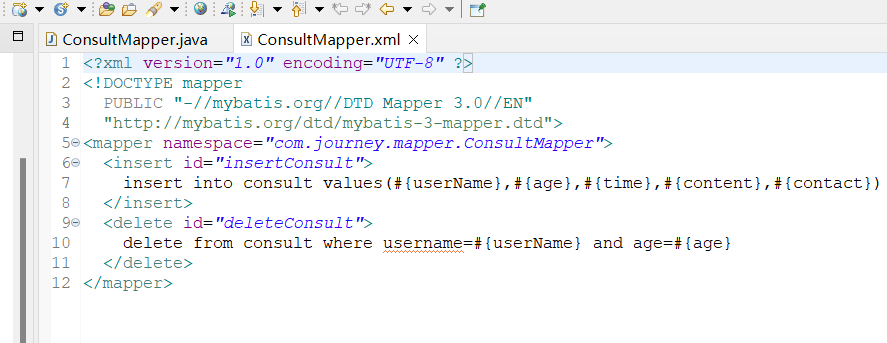
项目各目录包的示意图



首先是com.journey.mapper层，主要是写一些数据访问接口，是最接近数据库的一层，通过定义不同的方法接口，完成对数据库的不同操作



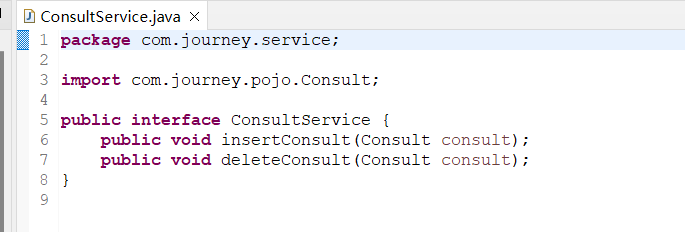
而这些对数据库进行实际操作的SQL语句，则通过Mybatis持久层框架，集中写在Resource（资源）目录下的xml文件中，通过映射的方式，实现不同接口类的对应方法



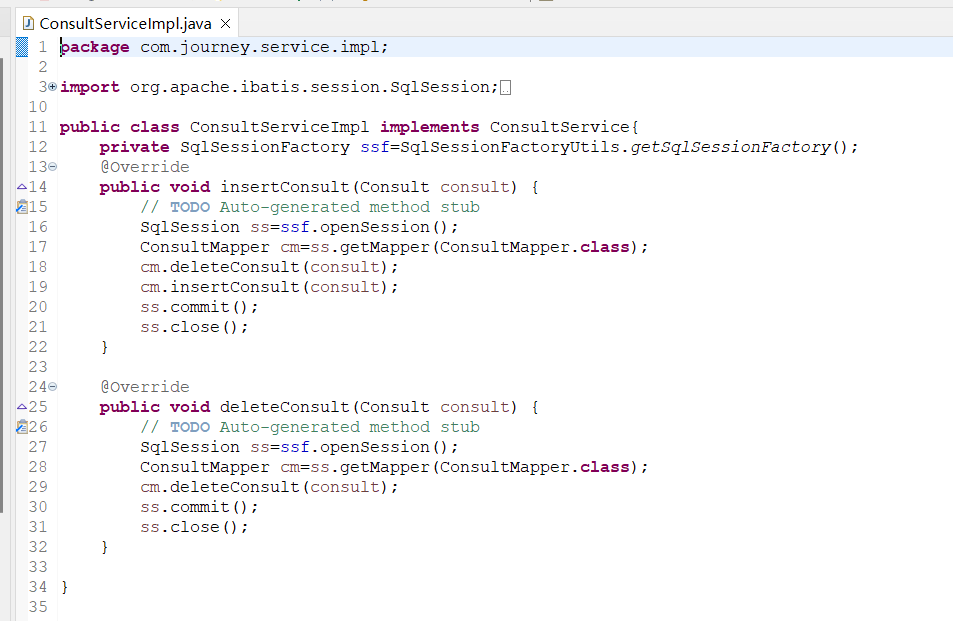
Com.journey.pojo包，定义各个普通的Java对象，也就我们常说的实体类，通常情况下，数据库的一个表对应一个实体类



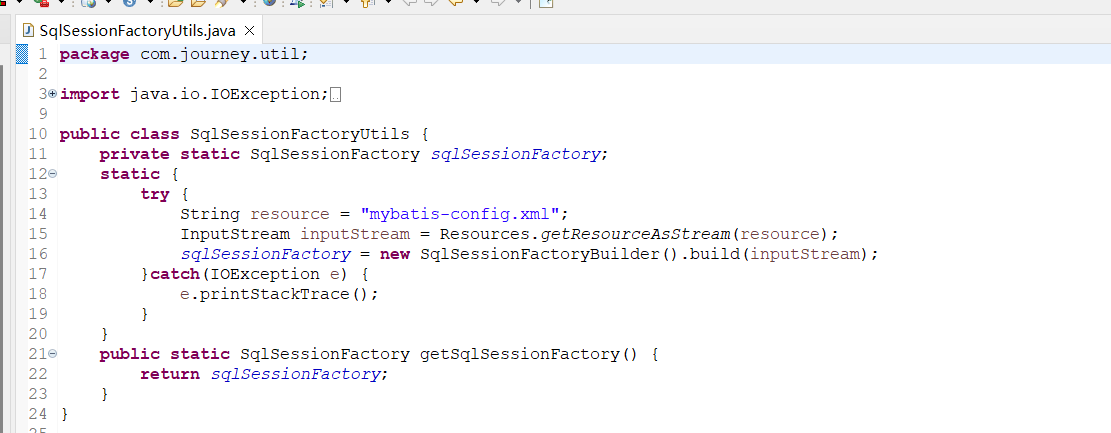
Com.journey.service层，用于定义不同的业务逻辑的接口，这一层主要是实现数据访问和业务逻辑的方法对接



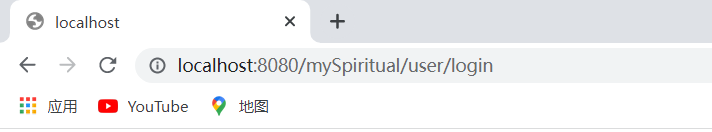
Com.journey.service.impl层，用于具体实现业务逻辑层的各个接口，在这一层编写实际的逻辑处理代码，如增删查改等操作

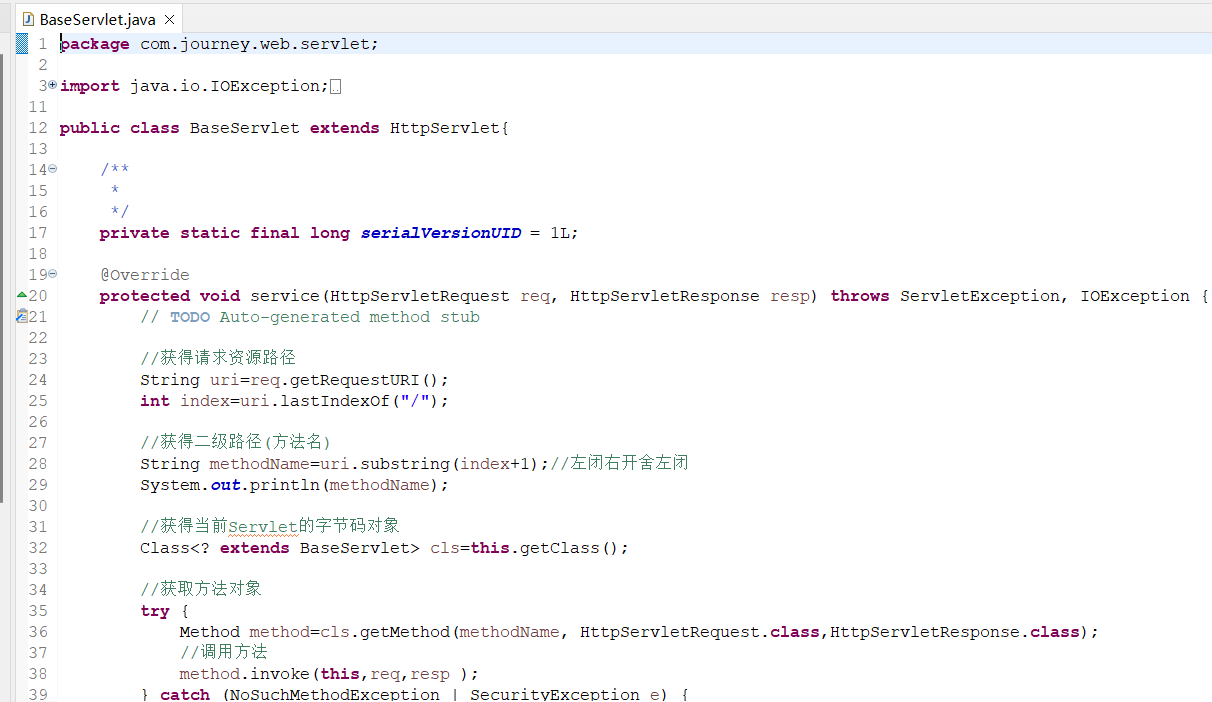


Com.journey.util层，这是为了实现高效地使用Mybatis的数据库连接池的功能，抽象出来的一个工具类，每一个业务逻辑层都可以使用该工具类，获得与数据库的连接

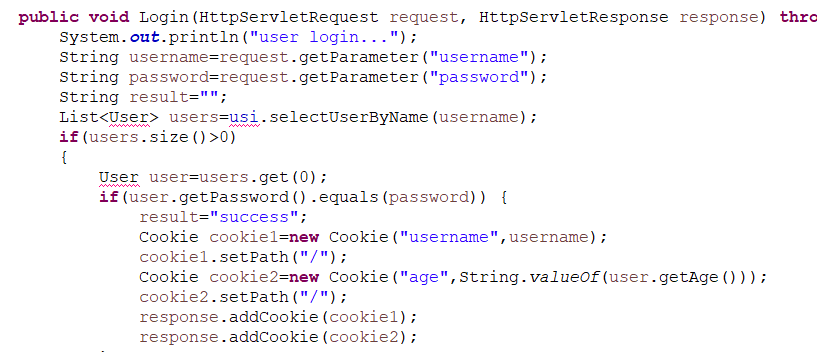


Com.journey.web.servlet层，就是我们常说的Controller控制器，是跟前端页面直接关联的，根据前端页面的不同交互操作，转发到不同的业务逻辑层，其实这里利用Java的一个重要特征，Java反射，通过继承一个BaseServlet，可以将请求的不同资源路径，转发到不同的servlet下的方法





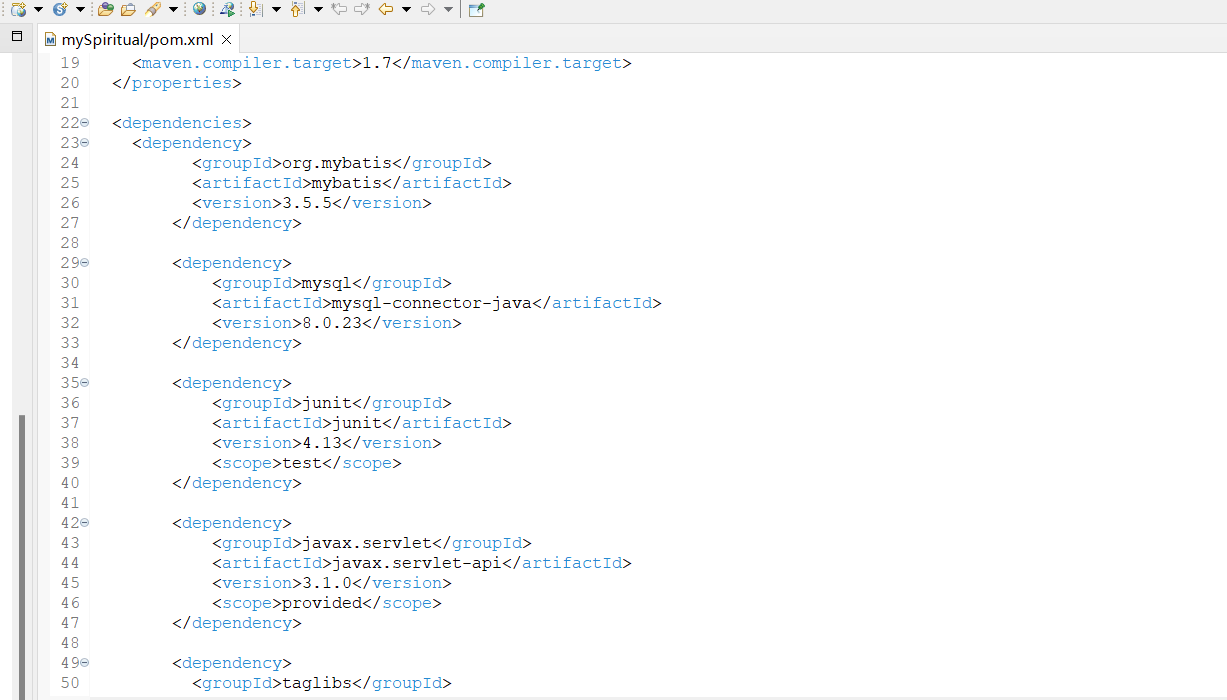
此外，为了记住当前登录的用户，我们使用了Cookie，来记录登录的用户的状态信息，如用户名和年龄，在登录验证成功时，我们把用户的用户名和年龄加入到当前浏览器的Cookie中，用于保存状态信息



Mybatis-config.xml，由于使用的是mybatis框架，进行数据库的连接和操作，在该配置文件里，配置数据库连接的信息，也就是数据库连接池，此外还对pojo包下的所有实体类定义了别名，简化了书写



Pom.xml文件是在使用Maven构建项目时，用于引入各个依赖的文件，对所有需要的jar依赖，都可以在该xml文件中引入



后端逻辑编写好了之后，考虑的就是前端页面如何进行请求数据，并且动态刷新，这里我们引入了Vue,js和axios.js实现请求数据和数据动态更新，

vue.js的最大特点就是数据双向绑定，当数据的值发生改变时，页面的值也会实时变化，

请求数据使用的是最通用的Ajax请求，异步数据请求，不需要刷新网页的同时动态更新数据，原生的Ajax的写法非常棘手，我们引入了axios异步框架，对ajax进行封装，简化书写

