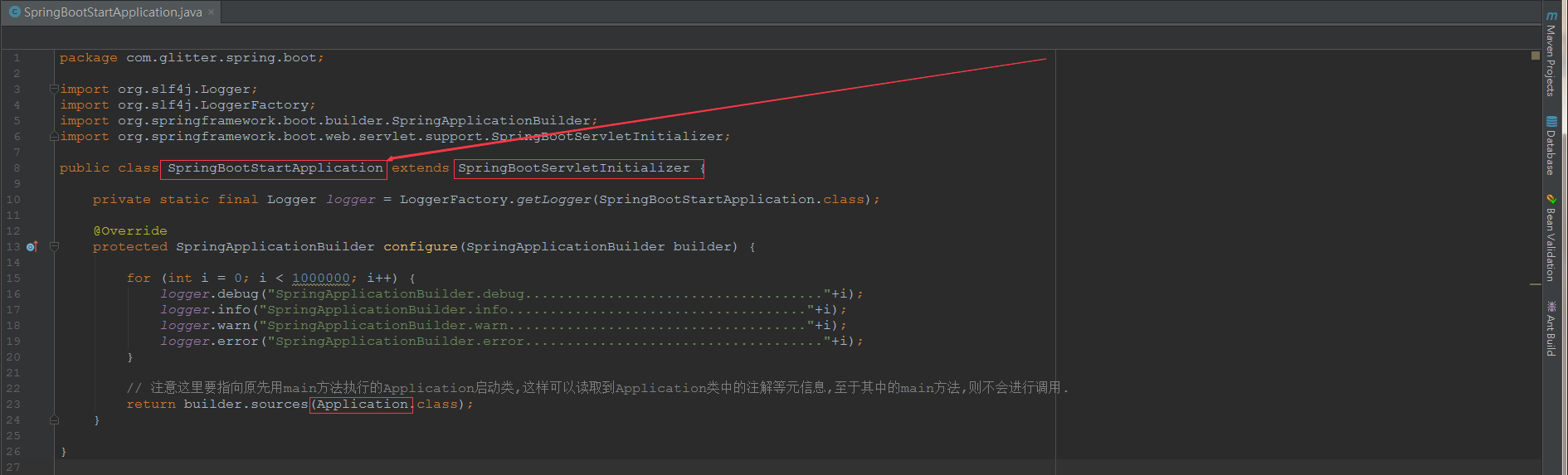
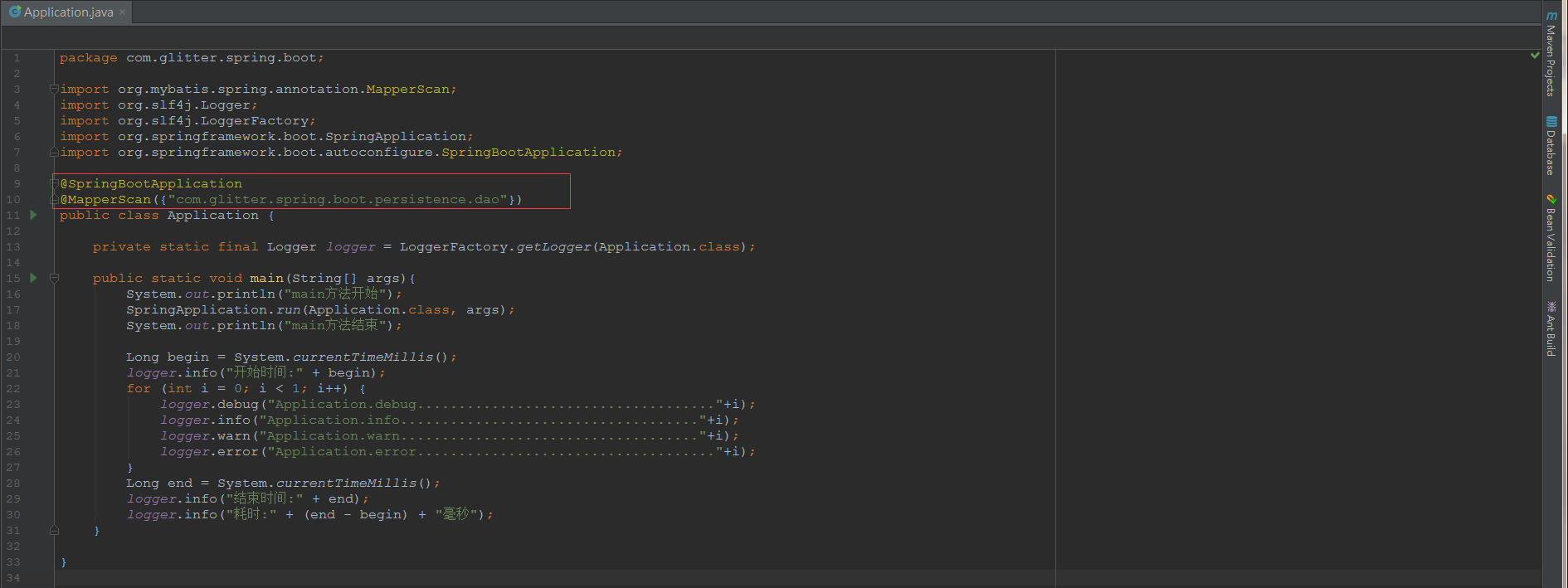
1.spring boot项目，在tomcat容器启动时，（tomcat8.0以上，支持servlet3.0api），springboot相关jar包利用servlet3.0支持spi机制，进行容器启动时的springboot初始化，初始化过程中会读取application.xml或application.yml文件的配置信息，

同时会把所有的配置过程都划归到注解中去（推荐），当然也支持xml。

springboot同时会加载项目中使用注解配置的类，比如@Coniguration注解，这就相当于是原来的配置文件，如果这个类实现了WebMvcConfigurer接口，就可以在其中配置很多Web特性的东西了，比如filter，servlet，listener，interceptor等，这就大体相当于原来的web.xml中做的事情了。

只是执行时机和执行方式发生了变化，web.xml是tomcat容器读取并进行相关的配置的。而这个类中filter，servlet，listener的是springboot利用servlet3.0支持spi和动态加载servlet、filter的api层面的支持，进行动态生成配置的。至于interceptor那是springboot自己的东西，不用多说了，想怎么玩怎么玩。







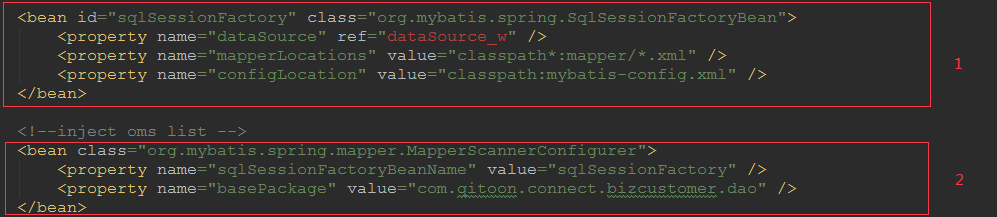
2. pom文件配置mybatis相关jar包依赖。

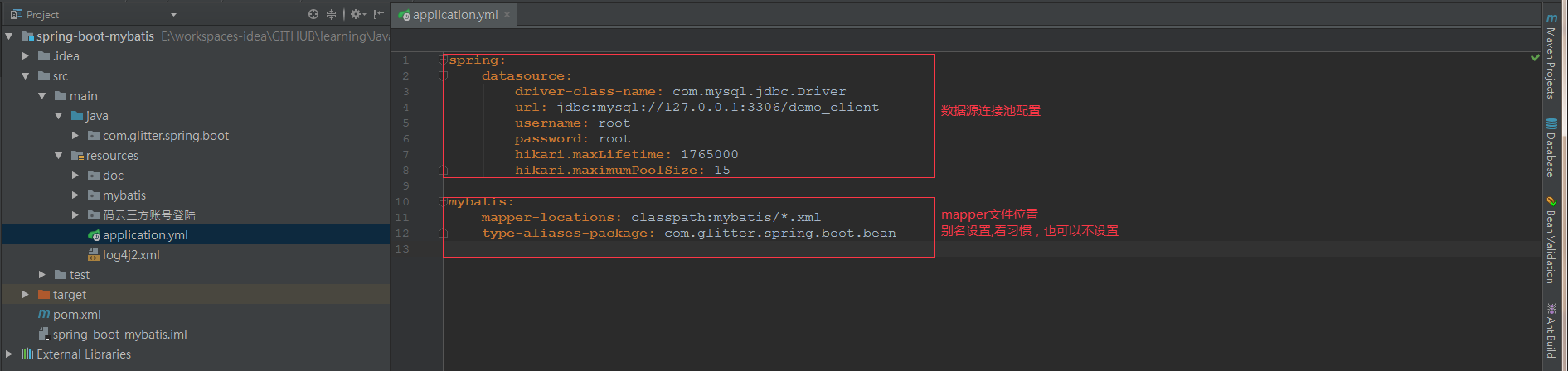
<!-- mybatis -->  
<dependency>  
 <groupId>org.mybatis.spring.boot</groupId>  
 <artifactId>mybatis-spring-boot-starter</artifactId>  
 <version>1.3.2</version>  
</dependency>  
<dependency>  
 <groupId>mysql</groupId>  
 <artifactId>mysql-connector-java</artifactId>  
 <version>5.1.47</version>  
</dependency>  
<!-- 分页插件,可以做一个自己的分页插件 -->  
<dependency>  
 <groupId>com.github.pagehelper</groupId>  
 <artifactId>pagehelper-spring-boot-starter</artifactId>  
 <version>1.2.5</version>  
</dependency>

3.数据源连接池配置和mybaitis的mapper文件路径配置，spring boot默认依赖hikri连接池jar包，并且在application.yml中如果没有特别指明spring.datasource.type类型属性的话，默认配置的就是使用hikari数据库连接池。

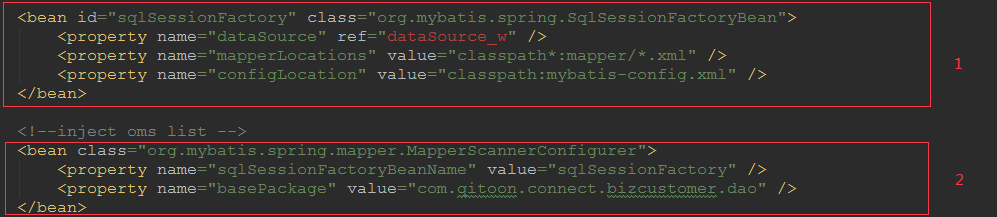
另外如果项目中只配置了一个数据源的话，mybatis的SqlSessionFactoryBean默认也是注入使用该数据源作为获取数据库连接的连接池使用的，并且SqlSessionFactoryBean对象不需要再特别申明就会被默认创建出来，当然也可以指定他要使用哪个数据源创建，不过那是多数据源的事情了。

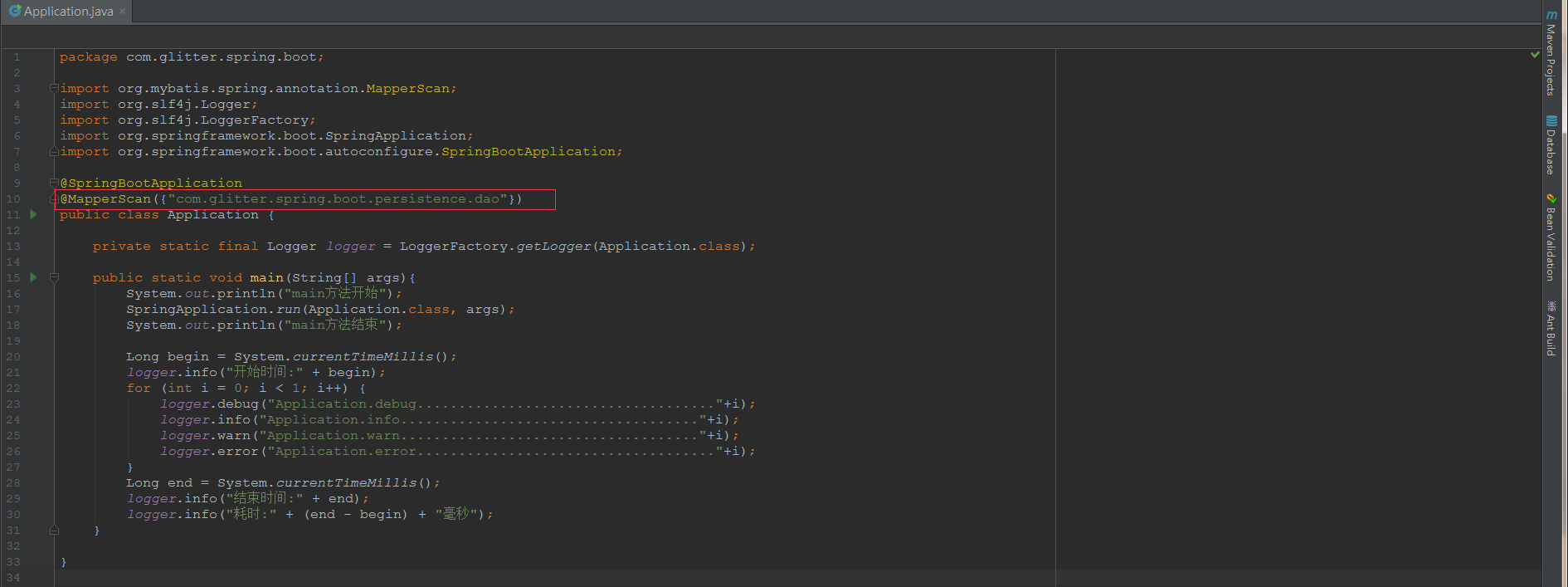
这相当于是传统中的第一步配置。



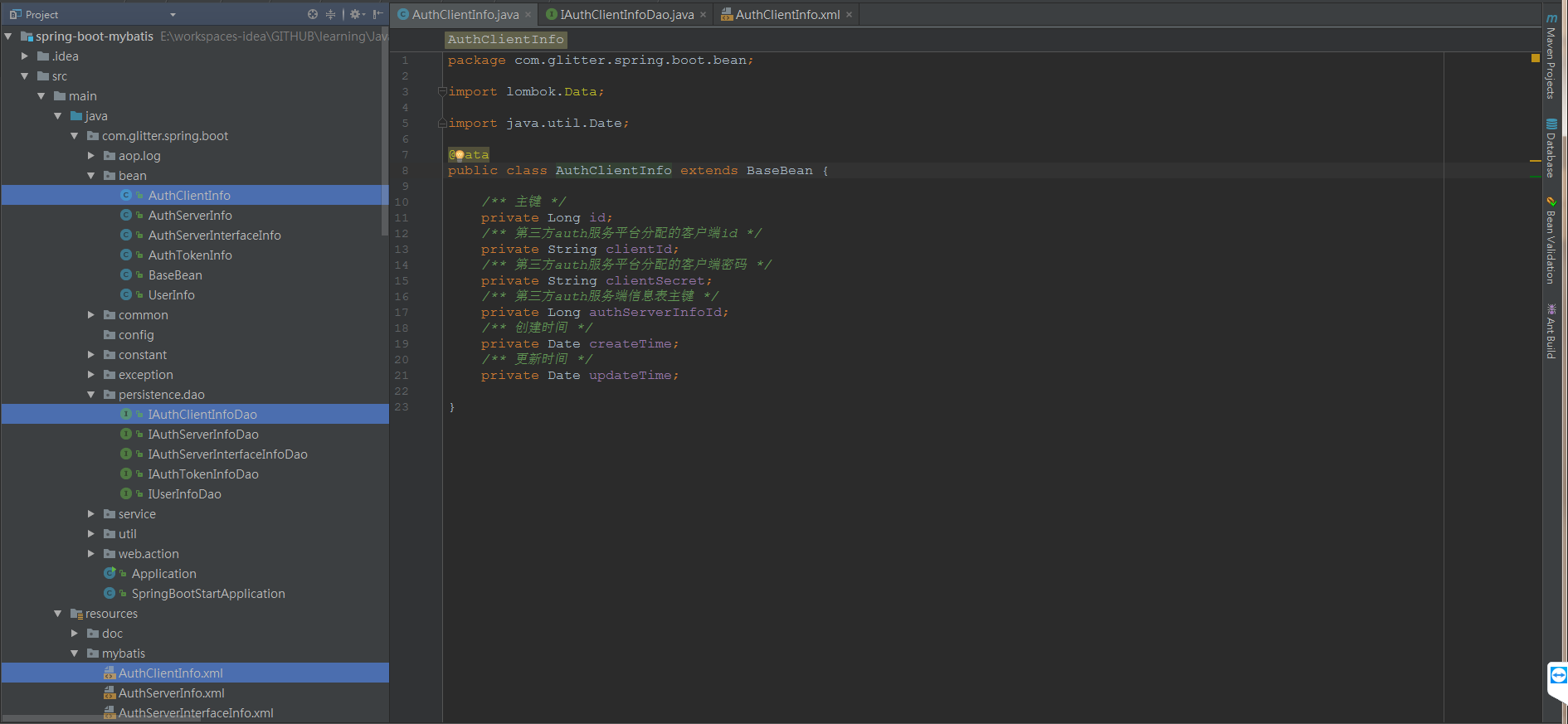


4.指明mybatis扫描的接口路径，背后是创建了MapperScannerConfigurer对象，并且该对象中的sqlSessionFactoryBean就是默认创建的出来的那个bean。相当于传统中的第2步。

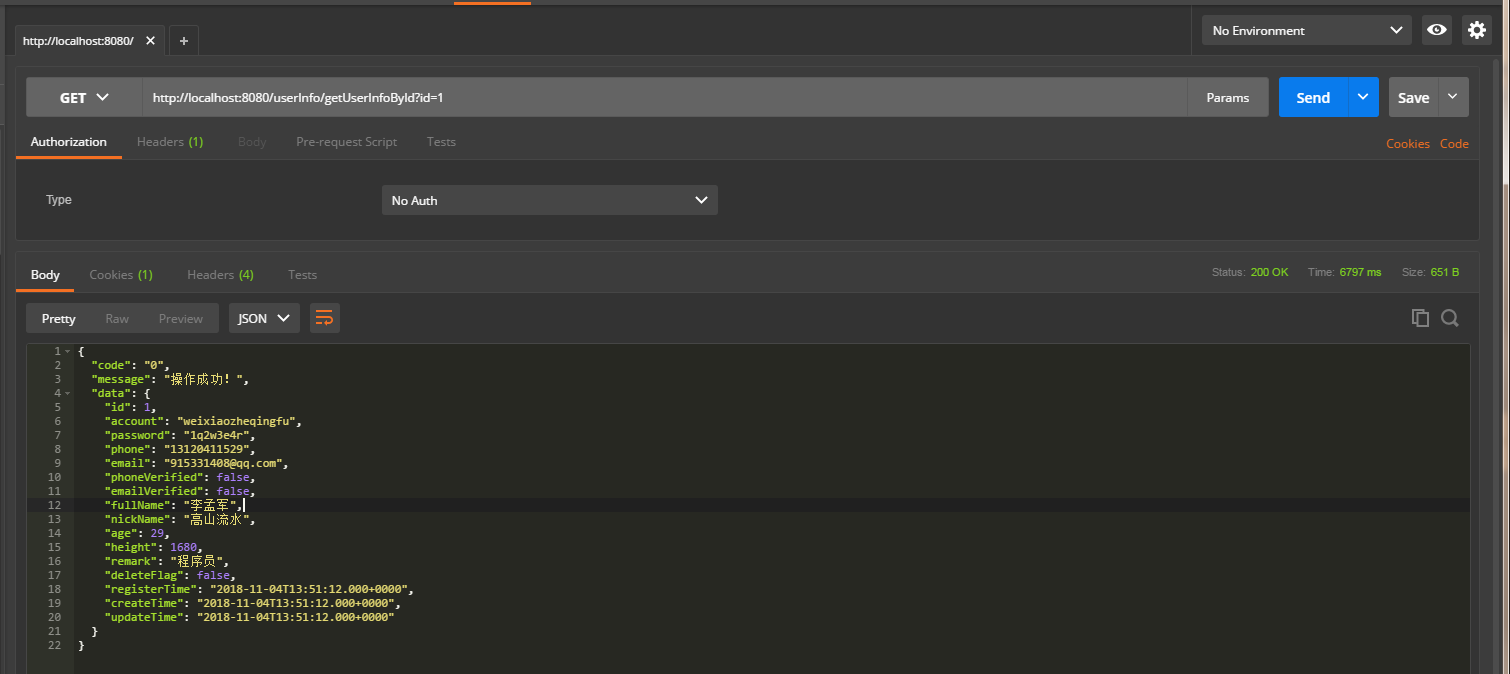




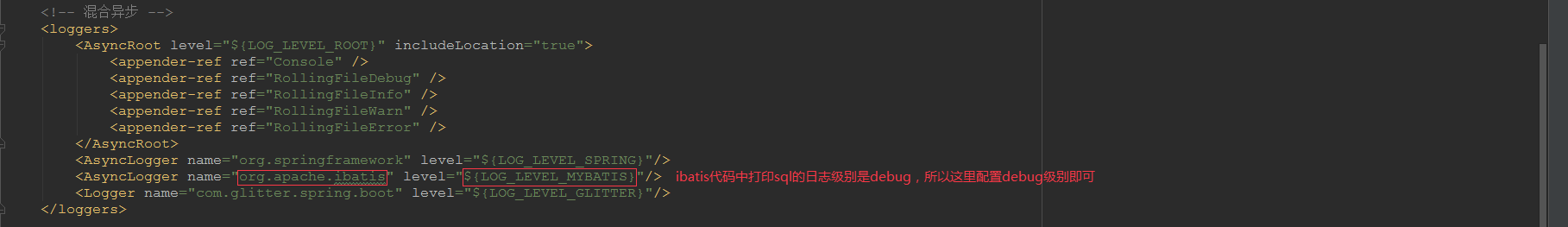
5.新建数据库，并反向生成对应的实体类，dao层接口类，mapper.xml文件。



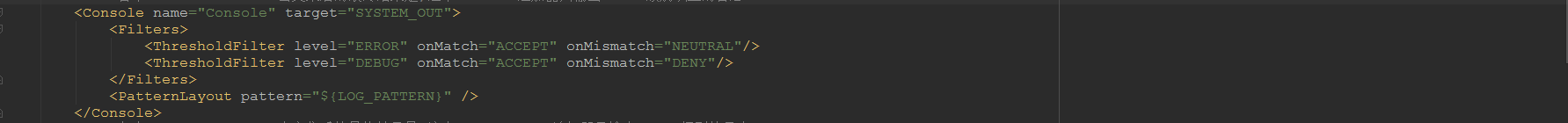
6.可以进行测试了。

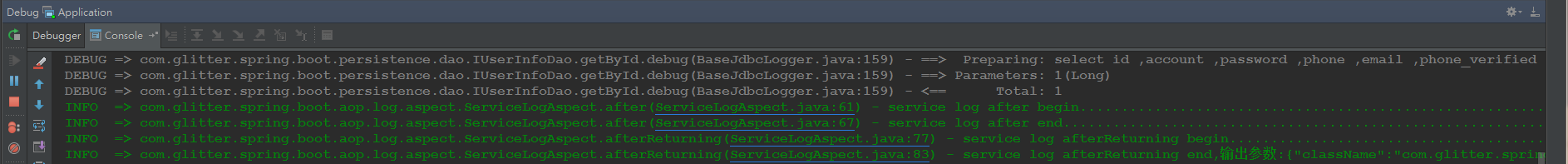


7.后台log4j2打印sql日志信息配置。



控制台如果对级别有过滤，要保证过滤后能打印DEBUG级别才行，如果没有配置过滤，那就算了，这样sql脚本就可以在控制台输出了。





8.分页插件的使用。

// TODO……

9.整个过程相当于传统的“1和2”步的组合，这种组合适用于小项目，因为通常只有一个数据源，且很难做读写数据源的分离。所以后续会继续写“1和3”步的组合在spring boot下是如何进行配置使用的，并且会进一步展开多数据源和读写分离以及mycat的使用。