**安全性**

1.注入问题

输入信息时对特殊字符如<script>标签等进行转义，防止 XSS注入；

在持久层的sql处理时对动态sql语句进行预编译，防止sql注入方式。

2.权限问题

使用安全框架（如shiro、springsecurity）对用户权限进行管理，用户无法访问超出自身权限范围的接口。

3.会话被窃取问题

用户在登录后，由服务端生成Token信息返回前端；对于需要登录或认证后才能访问的接口，发送请求时需要在请求头携带上这个Token认证信息，通过验证后才能访问，就算已认证后的会话被劫持，缺少Token信息，依然无法通过后端的拦截器。

4.数据库问题

防止用户直接操作数据库 用户只能通过给定的外部接口对数据库进行操作。

4.密码问题

后台管理员的账号密码储存在数据库里，密码传输前，前端进行简单的哈希加密，后端再进行哈希加盐加密。

**健壮性**

1.状态码

每次请求的响应结果都会向前端展示状态码，并加以文字信息描述，方便排查

解决问题。

如

｛

code：401  
 message:”无访问权限！！！”

｝

2.响应时间

对所有接口的异步响应设置超时时间，超时后抛出异常并返回对应状态码。

3.数据库访问

当查询、更新、插入等操作超过一定数目时抛出异常，并返回相应状态码，以保证数据库安全。

4.输入问题

对所有输入内容进行检测限制，对于超出输入范围或不符合输入要求的内容，进行提示或是异常处理

5.数据转换

对数据格式转换的处理，需要明确：数据转换前后的格式、转换规则、是否有数据在转换时需要特殊处理，如边界值等。

6.模块设计

对功能模块进行设计时，遵循高内聚低耦合的原则，保证各模块功能独立。