以常见的web应用程序开发中的登录认证界面的设计为例，通常的做法如下1-3。下面的作法有哪些安全问题？如何改进？你的web开发中d的登录界面通常还会使用哪种其他的方法？

* 1. web页面上用户输入账号、密码，点击登录
  2. 请求提交前，web端首先通过客户端脚本如javascript对密码原文进行md5处理
  3. 请求提交到后台，验证账号与密码是否与数据库中的一致，一致则认为登录成功，反之失败

窗体底端

**安全问题及解决方式：**

1. 如果使用http协议传输，HTTP是明文协议，因此，用户名和口令在网上也是明文发送的，这个很不安全。

解决： 要做到加密传输就必需使用HTTPS协议。也可以在注册账号表单在提交时候触发一个事件，这个事件的作用是把密码加密,这里可以推荐那个smd5插件,后端接收到密码后，再次加密存放到数据库中,因为不需要知道用户明文密码，所以不需要解密,当用户在登录页面提交表单时候,再次使用smd5加密，加密后再传输

1. **千万不要在cookie中存放用户的密码**。加密的密码都不行。因为这个密码可以被人获取并尝试离线穷举。
2. 如果密码很简单，如123456；这样的密码很容易被破解

解决：大部分网站会限制用户输入这些简单的容易破解的口令，比如必须含有字母、数字和下划线的组合

1. 记住密码中的安全问题：当你记住密码时，系统会生成一个cookie，cookie包括用户名和一个固定的散列值，这个固定的散列值一直使用。这样，你就可以在所有的设备和客户上都可以登录，而且可以有多个用户同时登录。这个并不是很安全。

解决：登录的时候，若是有“记住我”，那么，生成一个唯一的加密token，可以是用户名/过期时间/用户id/登录时间 等混合加密；在另外一张表维护；每次来，就判断用户名和对应的token，这个token就相当于一个临时的密码

**在web开发中还会在登陆中用到的其它方法：**

1. **多重认证**。比如：通过手机+邮件的方式让用户输入验证码。手机+邮件可能还不把握，因为手机要能会丢了，而我的手机可以访问我的邮箱。所
2. **使用验证码**。验证码是后台随机产生的一个短暂的验证码，这个验证码一般是一个计算机很难识别的图片。这样就可以防止以程序的方式来尝试用户的口令。
3. **用户口令失败次数**。两次口令尝试的间隔是5秒钟。三次以上错误，帐号被临时锁上30秒，5次以上帐号被锁1分钟，10次以上错误帐号被锁4小时……但是这会导致恶意用户用脚本来攻击，所以最好再加上验证码，验证码出错次数过多不禁止登录而是禁lP。