简版设计文档

1. **说明**

时间关系，本文档仅简单列举软件设计的一些要点，并非完整完善的设计文档。

1. **软件架构**
   1. 前端

前端拟采用LayUI框架，并设计响应式布局以适应PC端和移动端。

* 1. 后端

后端使用python编写，使用Flask框架，jinja2模板引擎，MySQL数据库，并使用DBUtils数据库工具建立数据库连接池。

拟采用flask-mail插件实现邮箱验证码的发送。

使用flask蓝图技术实现各子模块的独立编写，增加系统可扩展性。

1. **安全性设计**
   1. 数据安全

任何软件系统都存在漏洞被攻破的可能，为减少意外情况发生时的损失，保护数据安全，尽量减少数据库内存储的敏感信息。具体的有：

1. 不存储明文密码。数据库内所有账户密码均存储不可逆的md5值，不存储明文密码，即使数据泄露也不会泄露用户密码，为用户其他账号带来隐患。
   1. 登录流程

系统设计了登录流程：

1. 网页端登录，访问登录页面，服务器端返回的页面内隐含一个UUID（secret）及登录序号（secret\_key），APP端登录，先向系统后端请求该UUID（secret）及登录序号（secret\_key）；
2. 用户输入用户名密码，前端（APP端）计算密码的md5值，然后将md5值附加在UUID后面，形成的新的字符串，再计算md5值，将此值作为passwd\_md5参数发送回后端，同时发送的还有：用户名（username）、secret\_key（secret\_key），**注意，不得同时发送UUID（secret）**；
3. 网页端登录验证正确，向cookies写入token和新的secret，跳转到对应的页面，否则返回待错误信息的登录页面；
4. APP端验证正确，返回对应状态码（见API文档），否则亦然；
5. token用于弱安全性的网页端登录状态保持，随cookies一同发送，仅用于保持登录状态，其他各项权限校验、数字签名等皆使用secret，且secret不得与参数一同发送，否则它将失效。
   1. 可信化数字签名

调用API执行查询（GET）、添加（POST）、修改（POST）、删除（DELETE）等操作时，需要进行签名，签名的作用：

1. 验证用户身份，只允许权限范围内的操作；
2. 验证提交数据的可信度，防止他人而已篡改数据。

签名流程如下：

1. Secret不随请求发送（网页端将设置secret为secure以禁止在非HTTPS环境发送secret）；
2. 参数以键值对形式（也是默认type）发送，除原有参数外，还需发送keys、secret\_key和sign三个参数，其值分别为所有参数的key列表（英文逗号隔开）、secret对应的secret\_key（登录时返回的，注意，不是secret）和签名字符串；
3. 签名字符串的产生：
   1. 将所有的键值对参数（除sign、keys以外）以key=value&key2=value…的形式连接，然后在末尾加上secret，再计算md5，即为签名字符串；
   2. 拼接参数时，不需要加sign和keys这两个参数，因为拼接参数时它们还没有产生；
   3. 为避免secret泄漏，不得在请求中传输secret；
4. Keys的含义：keys是所有传输的参数的key的有序列表，记录产生签名字符串时的key的顺序，后端获得这个顺序才能以正确的顺序拼接参数验证签名，keys以英文逗号隔开；
   1. XSS攻击

所有提交的明文字符串数据都会进行HTML标签符号转译：

1. '<' => '&lt;'
2. '>' => '&gt;'
3. '"' => '&quot;'
4. ''' => '&#039;'

这样可以在一定程度上抵御xss攻击，对于API调用者，应当视情况考虑是否转译回原字符。

* 1. SQL注入

从请求获取的所有字符串参数，都会进行引号转译，以在一定程度上抵御SQL注入攻击，但系统使用者可能需要其他措施确保安全

* 1. 文件名处理

为避免通过覆盖系统文件的方式进行的攻击，上传的文件名将统一命名为：md5（孩子id+原始文件名），已存在的，将直接覆盖。