后端设计：

后端分两个类

User类和UserManager类

User类属于单个用户，可以登录，发帖和回帖。

UserManager类是User类的管理器

可以添加用户，随机用户发帖，随机用户顶贴等。

User类：

|  |  |  |  |
| --- | --- | --- | --- |
| class User | | | |
| self | username | string | 用户名 |
| password | string | 密码 |
| session | session | 登录状态 |
| method | login() | rtype: bool | 使用用户名密码登陆  返回登陆成功与否 |
| post(title, message) | title: string  message: string  rtype: bool | 使用session登陆  发表新帖子  返回发表成功与否 |
| reply(pid, message) | pid: string  message: string  rtype: bool | 使用session登陆  发表新回复  返回发表成功与否 |

UserMgr类

|  |  |  |  |
| --- | --- | --- | --- |
| class UserMgr | | | |
| self | users | <User>[] | 用户们 |
| method | getUserNum() | rtype: int | 读取users文件  一行一个以<TAB>分隔  用户名密码，#开头的行忽略  返回用户数量 |
| addUser(uname, pwd) | uname: string  pwd: string  rtype: int | 调用User.login()  登陆成功则将用户名密码写入目录下的users文件  注意用户是否重复  返回0：添加成功  1：用户名密码有误  2：重复添加的用户 |
| post(title, message) | title: string  message: string  rtype: bool | 调用User.post()  返回发表帖子成功与否 |
| preContent(type) | type: int  rtype: string | 根据type打开contents文件夹下的文件  返回文件内容 |
| writeContent(content) | content: string  rtype: None | 打开contents/confirm文件，写入content |
| up(pid, userNum) | pid: string  userNum: int  rtype: None | 打开users文件  读取相应数量的用户登录并回复contents/confirm文件中的内容，一条回复占一行，#开头的行忽略，\结尾的行是转义的回车，下一行是同一回复  循环用户，调用User.login()和User.reply()，并将返回的回复状态写入state文件，如（admin发表了“楼主说的好”：成功），每个回复占一行 |
| upShow() | rtype: string | 读取state文件，返回内容 |

界面设计：



界面分四部分：

左上角展示用户数量；

左下角发帖；

右上角添加用户；

右下角可以运行顶贴器。

* 状态刷新：

单击右下角的状态刷新，可以刷新用户数量和顶贴器回帖状态。

* 添加用户：

使用者输入用户名和密码，单击添加用户

* 发帖：

使用者输入帖子标题和内容，单击发表帖子（\*）

* 回帖：

第一步：使用者选择风格，单击预览

第二步：使用者编辑回帖内容，单击确认（\*）

第三部：使用者输入顶贴器使用的用户数量和帖子ID，单击运行

\* 由于tkinter的缺陷，输入框只能输一行，多行输入应使用输入按钮逐行输入。