|  |  |
| --- | --- |
| 验证身份（已经实现） | |
| 实现功能 | 用户提供名字，验证是否确有其人 |
| 请求体 | r'^confirm$' |
| 请求方式 | Post |
| 请求参数 | name:（用户的名字） |
| 返回格式 | JSON |
| 返回体格式 | {  status:0； (0代表操作成功，1代表操作失败)  no:用户的账号  } |

|  |  |
| --- | --- |
| 查询信息 | |
| 实现功能 | 查询图书馆的空位情况 |
| 请求体 | r'^search$' |
| 请求方式 | Get |
| 请求参数 | no:（用户的账号） |
| 返回格式 | JSON |
| 返回体格式 | {  status:0; (0代表操作成功且用户已经有了预约记录，1代表操作成功但用户没有预约记录，2代表操作失败)  message:[{ desk\_number:001 , floor:1},{ desk\_number:001 , floor:1}];（返回所有的空位信息）  rent:{desk\_number:001 , floor:1，start\_time,end\_time}（如果有预约记录，返回用户的预约记录，如果没有，返回null）  } |
| 如何实现 | 1. 遍历desk表，选出所有plug\_state为false的元组并返回 2. 遍历desk表，查找该用户是否预约过（有无student\_number的值为当前用户的值） |

|  |  |
| --- | --- |
| 预约座位 | |
| 实现功能 | 根据用户的需求预约座位 |
| 请求体 | r'^rent$' |
| 请求方式 | Post |
| 请求参数 | No:（用户的账号），desk\_number: (预约的座位号)  start\_time:(当前时间)，end\_time:（结束时间）  date |
| 返回格式 | JSON |
| 返回体格式 | {  status:0;（0：预约成功 ；1：输入数据有误 ；2：已有其他预约记录 ；3：已被他人预约；4：输入未完整）  desk\_information:[{desk\_id:1,floor:1,plug\_state:1}]  } |
| 如何实现 | 后端的同学：在Rent表中添加一条记录，修改表Desk中相关座位的信息如果用户已经有过预约，则什么都不干返回1 |

|  |  |
| --- | --- |
| 续约座位 | |
| 实现功能 | 根据用户的需求续约座位 |
| 请求体 | r'^renew$' |
| 请求方式 | Post |
| 请求参数 | no:（用户的账号），end\_time:（结束时间） |
| 返回格式 | JSON |
| 返回体格式 | {  status:0;（0代表续约成功，1代表预约失败，用户没有预约过，无法续约）  } |
| 如何实现 | 后端的同学：在Rent表中添加一条记录， |

|  |  |
| --- | --- |
| 取消预约 | |
| 实现功能 | 根据用户的需求取消座位 |
| 请求体 | r'^cancel$' |
| 请求方式 | Post |
| 请求参数 | no:（用户的账号）， |
| 返回格式 | JSON |
| 返回体格式 | {  status:0;（0代表取消成功，1代表取消失败，用户没有预约过，无法续约）  } |
| 如何实现 | 后端的同学：修改rent表中的相关记录（把end\_time 改为当前值） |