IGOR- Các hàm đề xuất của server

Dựa trên các tính năng của ứng dụng Igor, em/tôi viết một số đề xuất của các model và một số các controller:

1. ***Model***
   1. ***User***
2. ***Create\_user(user details): include openID, accept from google***
3. ***Get\_user\_details(user\_id)***
4. ***Get\_users\_by\_subject(subject\_id)***
5. ***Get\_users\_by\_class(class\_id)***
6. ***Get\_user\_by\_name(string name)***
7. ***Get\_users\_in\_project(class\_id)***
8. ***Get\_users\_same\_project(user\_id)***
   1. ***Subject***
9. ***Get\_all\_subject***
10. ***Get\_subject\_detail(subject\_id)***
    1. ***Class\_subject***
11. ***Get\_classes\_by\_term(term=current)***
12. ***Get\_classes\_by\_subject(subject\_id, term=current)***
13. ***Get\_classes\_by\_user(user\_id, term=current)***
14. ***Get\_classes\_detail(class\_id)***
    1. ***Scheduler***
15. ***Get\_scheduler(user\_id, int term=current)***
16. ***Get\_scheduler\_by\_time(user\_id, day\_of\_week, term=current)***
17. ***Add\_scheduler(user\_id, class\_id) (term=current)***
18. ***Delete\_scheduler(user\_id, class\_id)***
    1. ***Notification***
19. ***Get\_all\_notifications(owner, type=new)***

Type

|  |  |  |
| --- | --- | --- |
| Type |  |  |
| 0 | all | Get all notifications |
| 1 | New (default) | Get all new notifications |

1. ***Mark\_read(notifications)***
2. ***Add\_new(notification\_detail)***
   1. ***Jobs***
3. ***Get\_jobs(int type, value)***

|  |  |  |
| --- | --- | --- |
| Type |  |  |
| 0 | all |  |
| 1 | unfinished |  |
| 2 | finished |  |

If value is empty then get all jobs

Else get job with name like value

1. ***Get\_job\_by\_class(user\_id, class\_id)***
2. ***Get\_job\_detail(job\_id)***
3. ***Add\_class\_job(user\_id, job\_detail, class\_id) (test in here)***
4. ***Delete\_job(job\_id)***
5. ***Mark\_job\_finished(job\_id)***
6. ***Mark\_jobs\_finished(job list)***
7. ***Controller (service)***
   1. ***User***
8. ***Register(user details)***

Give all user detail, email and user open\_id from gmail to the registration.

***Pseudo Code:***

|  |
| --- |
| Call User\_model.create\_user(user details)  Return user\_id |

1. ***Login(email, password) – user open\_id***

Input: User email, password for gmail authentication

Output: login status (true, false). If true, send session id to the mobile

***Pseudo Code:***

|  |
| --- |
| User\_model.get\_user\_details\_by\_email(open |

1. ***Add\_friend(user\_id, friend\_id)***

Add friend to user

Input: user\_id: user want to add new friend, friend\_id)

Output: return status of add\_friend operator, friend details

1. ***Is\_friend(user1\_id, user2 \_id)***

Check 2 user is friend or not

Input: two user id

Output: true or false. (optional: details of 2 user)

1. ***Search\_user(type, string = emtpy) - OK***

Input

Type

|  |  |  |
| --- | --- | --- |
| Value | Operator | Note |
| 0: all | Get all users in system |  |
| 1: same\_group | Get all users have same class |  |
| 2: by\_class\_id | Get all user by class\_id | optional |
| 2: same\_subject | Get all user have same subject |  |
| 3: by\_subject\_id | Get all user by subject\_id | optional |
| 4: student\_code | Get user by identified code |  |
| 5: by\_student\_name |  |  |
| 4: mutual\_friend | Get all user have mutual friend | optional |

String is value depend on search type

Output: Return is the list of user (details is optional)

***Pseudo Code:***

|  |
| --- |
| Switch case (type):  2: user\_model.get\_user\_by\_class(value);  3: user\_model.get\_user\_by\_subject(value);  5: user\_model.get\_user\_by\_name(name); |

1. ***Get\_user\_detail(user\_id) - OK***

Input: id key of user

Output: the arrary details of user

1. ***Get\_user\_by\_project(class\_id)***

***Pseudo Code:***

|  |
| --- |
| User\_model.get\_users\_in\_project(class\_id) |

* 1. ***Subjects***

1. ***Get\_all\_subjects 🡪 optional***

Input: none

Output: list all subjects

***Pseudo Code:***

|  |
| --- |
| Subject\_model.get\_all\_subjects |

1. ***Get\_subjects\_by\_user(user\_id)***

Input: user\_id

Output: list all user’s subjects

1. ***Add\_subject\_to\_user(subject, user) 🡪 optional***

Input: user\_id

Output: status of operation (option: subject details and user details

1. ***Get\_subject\_detail(subject\_id)***

Input: subject id

Output: all subject details

***Pseudo Code:***

|  |
| --- |
| Subject\_model.get\_subject\_detail(subject\_id) |

* 1. ***Class\_subjects***

1. ***Get\_all\_classes 🡪 optional***

Input: none

Output: list all classes

1. ***Get\_class\_by\_subject(subject\_id, term=current) - OK***

Input: subject\_id

Output: all subject’s class

***Pseudo Code:***

|  |
| --- |
| Class\_subject\_model.get\_class\_by\_subject(subject\_id, term=current) |

1. ***Get\_classes\_by\_user(user\_id, term=current) - OK***

***Pseudo Code:***

|  |
| --- |
| Class\_subject \_model.get\_class\_by\_user(user\_id, term=current) |

1. ***Get\_class\_detail(class\_id) - OK***
2. ***Add\_class\_to\_user(class, user)***

***Pseudo Code:***

|  |
| --- |
| Check\_scheduler if class is same period with other class  Scheduler\_model.Add\_scheduler(class\_id, class\_id, term=current) |

1. ***Change\_class(user\_id, old\_class\_id, new\_class\_id)***

***Pseudo Code:***

|  |
| --- |
| Scheduler\_model.Add\_scheduler(class\_id, class\_id, term=current)  Scheduler\_mode.Delete\_scheduler(user\_id, class\_id, term=current) |

1. ***Delete\_class (user\_id, class\_id) - OK***

***Pseudo Code:***

|  |
| --- |
| Scheduler\_mode.Delete\_scheduler(user\_id, class\_id) |

* 1. ***Scheduler***

1. ***Get\_scheduler(user\_id, int term=current)***

Get user’s scheduler

Input: user\_id, term (default=current). (ex: term=20071, 20072, 20073..)

Output: scheduler of term

***Pseudo Code:***

|  |
| --- |
| Scheduler\_mode.Get\_scheduler(user\_id, term=current) |

1. ***Get\_scheduler\_by\_time(user\_id, day\_of\_week, term)***

Get user’s scheduler by day

Output: scheduler in specific time

***Pseudo Code:***

|  |
| --- |
| Scheduler\_mode.Get\_scheduler\_by\_time(user\_id, term=current) |

1. ***Add\_class\_to\_scheduler(owner, class, day\_of\_week, period, term)***
2. ***Update\_scheduler(scheluder\_id, …)***
3. ***Delete\_scheduler(scheduler)***
   1. ***Notification***
4. ***Get\_all\_notification(owner,type=new, offset=0)***

***Pseudo Code:***

|  |
| --- |
| Notification\_model.Get\_all\_notification(owner, type=new) |

1. ***Push\_notification***
2. ***Mark\_read(notifications)***

***Pseudo Code:***

|  |
| --- |
| Notification\_model.Mark\_read(notifications) |

1. ***Get\_notification\_details(notification)***
   1. ***Projects***
2. ***Get\_projects(user\_id, int term=current)***
3. ***Get\_project\_detail(project\_id)***
4. ***Get\_projects\_by\_class(class\_id)***
5. ***Get\_project\_by\_user\_class(user\_id, class\_id)***
   1. ***Jobs***
6. ***Get\_jobs\_by\_class\_id(user\_id, class\_id)***
7. ***Get\_unwork\_jobs(user\_id)***
8. ***Get\_job\_by\_time(user\_id, from=today,to=today)***

Input: user id, from\_time=1,2,3,4(default = today – 0h), to\_time (default=today – 12h)

Output: list of job in specific time

1. ***Get\_job\_detail(job\_id)***
2. ***Add\_personal\_job(user\_id, job\_detail)***
3. ***Add\_class\_job(user\_id, job\_detail, class\_id)***
4. ***Delete\_job(job\_id)***
5. ***Mark\_job\_finished(job\_id)***
6. ***Mark\_job\_finished(job list)***
7. ***Share\_job(job\_id, owner, to\_user)***
8. ***Accept\_job(job\_id, owner, from\_user)***

***Pseudo Code:***

|  |
| --- |
| Check notification, ensure notification was marked  Job\_model.Add\_class\_job(user\_id, job\_detail, class\_id) |

1. ***Share\_job\_to\_class(job\_id, from\_user\_id, class\_id)***

***Pseudo Code:***

|  |
| --- |
| Notification\_model(add notification to all user in class) |

1. ***Share\_job\_to\_project(job\_id, from\_user\_id, project\_id)***

***Pseudo Code:***

|  |
| --- |
| Notification\_model(add notification to all user in project) |

1. ***Ultility Modules***
   1. ***Time\_module***
2. ***Get\_time\_string***

Output: time in string form

|  |  |
| --- | --- |
| < 60 seconds | Xxx seconds ago |
| < 60 minutes | Xxx minutes |
| < 24 hour | Xxx hours yyy minutes ago |
| < 7 days | Xxx (>7) days ago |
| > 7 days | Date xxx month yyyy year xxxx |

1. ***Make\_time(year, month, day, hour, minute, second)***

* Call python time function

Output: time in unix form

1. ***Convert\_time\_to\_utc(time\_zone, time)***

Output: time in utc form.

* 1. ***Phrase\_module***