LiMen API Specification

2. Web service for listing of the existing fans

This API is used for listing corresponding user’s friend list member

* URI: /fan/list
* Method: POST
* Content-Type: application/json
* Accept: application/json
* Input Format: JSON
* Output Format : JSON

|  |  |  |  |
| --- | --- | --- | --- |
| **Input Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| userId | Integer |  | The user Id of the friend list owner |

|  |  |  |  |
| --- | --- | --- | --- |
| **Output Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| userFanList | Array of object | {fan Object} |  |
| {fan Object}- fanRealName | String |  |  |
| {fan Object}- fanUserId | Integer |  |  |
| {fan Object}- fanStatus | Integer |  | Fan status,  64 = ignored friend  32 = awaiting approval  8 = list owner  4 = power member, not available in friend list  2 = friend  1 = invited to join |

**Sample:**

**Request:**

{

"userId": 11

}

**Response:**

{

"userFanList": [{

"fanRealName": "Jacob Yu",

"fanUserId": 12,

"fanStatus": 1

},

{

"fanRealName": "Ethan Lu",

"fanUserId": 13,

"fanStatus": 64

},

{

"fanRealName": "Michael Pui",

"fanUserId": 14,

"fanStatus": 1

}]

}

3. Web service for viewing of the existing fans

This API is used for viewing corresponding user’s details

* URI: /fan/view
* Input and Output Format: JSON
* Method: POST
* Content-Type: application/json
* Accept: application/json
* Input Format: JSON
* Output Format: JSON

|  |  |  |  |
| --- | --- | --- | --- |
| **Input Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| userId | Integer |  | The user Id of friend |

|  |  |  |  |
| --- | --- | --- | --- |
| **Output Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| fanUserId | Integer |  |  |
| realName | String |  |  |
| photo | String |  |  |
| dateOfBirth | Date |  |  |

**Sample:**

**Request:**

{

"userId": 11

}

**Response:**

{

"fanUserId": 11,

"realName": "Alison Wu",

"photo": null,

"dateOfBirth": "1993-05-16"

}

4. Web service for adding of the new fans

This API is used for adding corresponding users into friend list

* URI: /fan/add
* Method: POST
* Content-Type: application/json
* Accept: application/json
* Input Format: JSON
* Output Format: JSON

|  |  |  |  |
| --- | --- | --- | --- |
| **Input Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| userId | Integer |  | The user Id of friend list owner |
| userIdList | Array of Integer |  | User Ids to be friends |

|  |  |  |  |
| --- | --- | --- | --- |
| **Output Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| responseStatus | String | success/fail |  |

**Sample:**

**Request:**

{

"userId": 11,

"userIdList": [12, 13,14]

}

**Response:**

{

"responseStatus": "success"

}

5. Web service for changing of status for the fans

This API is used for updating corresponding user member status in friend list

* URI: /fan/update
* Method: POST
* Content-Type: application/json
* Accept: application/json
* Input Format: JSON
* Output Format: JSON

|  |  |  |  |
| --- | --- | --- | --- |
| **Input Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| userId | Integer |  | The user Id of friend list owner |
| userIdList | Array of Integer |  | Fan’s user Ids |
| status | Integer |  | Status value to be updated,   64 = ignored friend  32 = awaiting approval  8 = list owner  4 = power member, not available in friend list  2 = friend  1 = invited to join |

|  |  |  |  |
| --- | --- | --- | --- |
| **Output Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| responseStatus | String | success/fail |  |

**Sample:**

**Request:**

{

"userId": 11,

"userIdList": [12, 13],

"status": 2

}

**Response:**

{

"responseStatus": "success"

}

6. Web service for deleting the existing fans

This API is used for deleting corresponding user member in friend list

* URI: /fan/delete
* Method: POST
* Content-Type: application/json
* Accept: application/json
* Input Format: JSON
* Output Format: JSON

|  |  |  |  |
| --- | --- | --- | --- |
| **Input Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| userId | Integer |  | The user Id of friend list owner |
| userIdList | Array of Integer |  | Fan’s user Ids |

|  |  |  |  |
| --- | --- | --- | --- |
| **Output Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| responseStatus | String | success/fail |  |

**Sample:**

**Request:**

{

"userId": 11,

"userIdList": [13]

}

**Response:**

{

"responseStatus": "success"

}

7. Web service for editing the exiting fans

Confirmed by Kuen’s email on 28 February 2013, this item is ignored.

8. Web service for sending the message within groups or individual users

This API is used for sending message for the request of the friend invitation

* URI: /sendAddFriendMessage
* Method: POST
* Content-Type: application/json
* Accept: application/json
* Input Format: JSON
* Output Format: JSON

|  |  |  |  |
| --- | --- | --- | --- |
| **Input Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| fromUserId | Integer |  | The user Id of the sender |
| toUserId | Integer |  | The user Id of the receiver |
| Content | String |  | The message content |

|  |  |  |  |
| --- | --- | --- | --- |
| **Output Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| resultCode | String | 999/200 | 999: Failure, 200: Success |
| Description | String | SUCCESS  INTERNAL\_ERROR  TARGET\_NOT\_IN\_PENDING\_FD\_LIST |  |

**Sample:**

**Request:**

{

“fromUserId”:”1”,

“toUserId”:”4”,

“content”:”Please accept my invitation!”

}

**Response:**

{

“resultCode”:”200”.

“description”:”SUCCESS”

}

9. Web service for searching the particular users, including user name, email and group

This API is used for searching users by user name, email and group

* URI: /search
* Method: POST
* Content-Type: application/json
* Accept: application/json
* Input Format: JSON
* Output Format: Array of JSON

|  |  |  |  |
| --- | --- | --- | --- |
| **Input Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| Keyword | String |  | The search keyword |
| Type | String | user  email  group | Search By User Name  Search By Email  Search By Group Name |
| userId | Integer |  | The user Id of the searcher |

|  |  |  |  |
| --- | --- | --- | --- |
| **Output Parameters** | | | |
| **Field Name** | **Type** | **Range/Value** | **Remarks** |
| Id | Integer |  | User ID |
| userName | String |  | User Name |
| status | Byte | 0 | This field will not be extracted from Database |
| password | String | null | This field will not be extracted from Database |
| schoolId | Integer | null | This field will not be extracted from Database |
| createTime | Timestamp | null | This field will not be extracted from Database |

**Sample:**

**Request:**

{

“keyword”:”test”,

“type”:”username”,

“userId”:”1”

}

**Response:**

[

{

“id”:”1”.

“status”:0,

“password”:null,

“userName”:”test1”,

“schoolID”:null,

“createTime”:null

},

{

“id”:”2”.

“status”:0,

“password”:null,

“userName”:”test2”,

“schoolID”:null,

“createTime”:null

}

]