Interface

Code Request Summary

The server deciphers each request through a series of codes:

DEFAULT\_USER\_LEVEL = 0;

SERVER\_PORT = 80;

REGISTER\_ACTION = 100;

LOGIN\_ACTION = 101;

MESSAGE\_ACTION = 102;

LOGIN\_SUCCESS\_CODE = 200;

LOGIN\_USER\_NON\_EXIST\_CODE = 201;

LOGIN\_EXIST\_CODE = 202;

RESISTER\_SUCCESS = 300;

REGISTER\_FAIL = 301;

FRIEND\_GET = 500;

FRIEND\_GET\_FAIL = 501;

FRIEND\_GET\_SUCCESS = 502;

FRIEND\_SEARCH = 503;

FRIEND\_SEARCH\_FAIL = 504;

FRIEND\_SEARCH\_SUCCESS = 505;

FRIEND\_ADD = 506;

FRIEND\_ADD\_FAIL = 507;

FRIEND\_ADD\_SUCCESS = 508;

FRIEND\_CHECK = 509;

FRIEND\_CHECK\_FAIL = 510;

FRIEND\_CHECK\_SUCCESS = 511;

# User Online Checking Service

Check whether the user is online.

Request

{

Username: string

action: int

}

Return

{

Status: Boolean

Code: int

}

# Add Friend Service

Request

{

Username: string

action: int

}

Return

{

Status: Boolean

Code: int

}

# Search User Service

Request

{

Username: string

action: int

}

Return

{

Status: Boolean

Code: int

Result: array

}

Fetch Friend List Request Service

The interface between the server and client which allows clients to ask the server to send a JSON list of all the user’s friends. The friends’ online status is not available yet.

Request

{

action: int

}

Return

{

Status: Boolean

Code: int

Result: array

}

Example Request

Send

{

“action”: 500

}

Return

{

"status":"success",

"code":502,

"result":[

{"username":"1"},

{"username":"3"},

{"username":"admin3"},

{"username":"pls"},

{"username":"serious"}

]

}

Server Login Service

The interface between the server and client which allows clients send login request to server.

When user logs in successfully, the user will be assigned a token, which is a unique ID of current session. Whenever the user sends an action to the server, the token must be attached as the authentication.

Message

{

username: string,

password: string,

action: int

}

Return

{

type: string,

code: int,

}

Example Request

Send

{

“username”: “test”,

“password”: “donttellothers”,

“action”: 101

}

Return

{

"type": "success",

"code": 200,

}

Server Messaging Service

The interface between the server and client which allows clients send message to others.

The user must be assigned a token first, or log in will be failed.

Message

{

action: int

to: string

message: string

}

Return

{

type: string,

code: int,

}

Example Request

Send

{

“action”: 201

“to”: “zirenxiao”

“message”: “have you seen my beer Tibbers?”

}

Return

{

"type": "success",

"code": 200,

}

Client Register Service

The interface between the server and client which allows clients to create a new account in database.

Message

{

username: string,

password: string,

email: string,

name: string,

action: int

}

Return

{

type: string,

code: int,

}

Example Request

Send

{

“username”: “test”,

“password”: “donttellothers”,

“email”: “[auser@catchmeifyoucan.com](mailto:auser@catchmeifyoucan.com)”,

“name”: “tester”,

“action”: 100

}

Return

{

"type": "success",

"code": 201

}