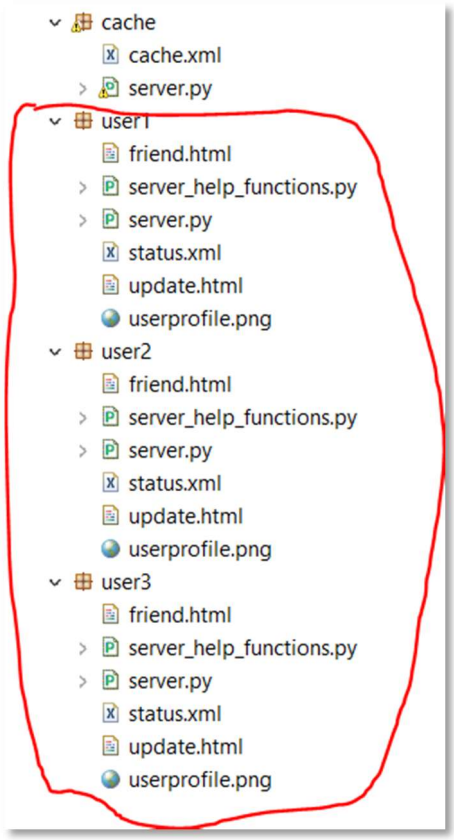
Student ID:17272381

Name: Zhihan Wang

1. File Framework

1.1 Users files

There are 3 x same file structure folders for different users that have been created for demonstration and test purpose with the file description as below, each user has almost exact same code:



status.xml – storage of each user's friend list and status data.

Data	Comments	Examples
<data ipAddress="" userID="">	overall data tag with userID and ipAddress Attributes	<data ipAddress="http://localhost:8080" userID="user1">
<friendlist>	User's friendlist IP address	
</ IPaddress>	Each individual IP address in this user	<friendlist> <IPaddress>http://localhost:8081</IPaddress> <IPaddress>http://localhost:8082</IPaddress></friendlist>
</friendlist>		

<status>	Use's each posted status data	
</lastModified>	The last modfied timestamp after user add a status or friend has liked this status	<lastModified>Sun, 14 Apr 2019 18:09:06 GMT</lastModified>
</ timestamp>	The timestamp that user post this status	<timestamp>Sun, 14 Apr 2019 18:07:10 GMT</timestamp>
</ content >	content of this status text: max 200 characters controlled by browser text box	<content>hello world</content>
</likes>	a unique list of IP addresses from IP address of friend list who has liked this status	
</ IPaddress>	IP addresses from friendlist IP	<likes> <IPaddress>http://localhost:8081</IPaddress> <IPaddress>http://localhost:8082</IPaddress> </likes>
</likes>		
</status>		
</data>		

update.html – display user's existing status, user profile picture, friends.html link, post and reset status function.

friend.html – display user friends' status, profile picture, like button and number of likes received for each status.

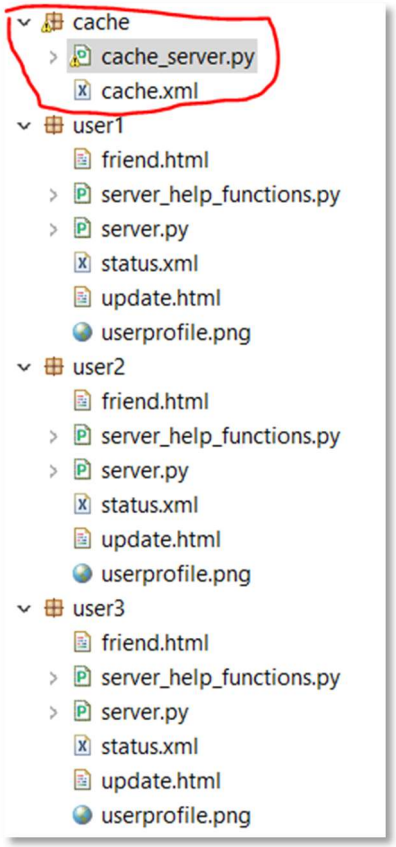
server.py – python server script to handle requests from client

server_help_functions.py – help functions to implement server side operations.

userprofile.png – user profile picture.

1.2 Cache files

cache files have been used to store time and date for all the user's latest status update and status that has been liked.



cache.xml – storage of all users' latest time and date for their status data by their own status update or status liked by their friends.

Data	**Comments**	**Examples**
<data ipAddress="" userID="">	overall data tag with userID and ipAddress Attributes	<data ipAddress="http://localhost:8080" userID="cache">
<user userID="">	Each user data	<user userID="user1">
</ IPaddress >	This user's ip address	<IPaddress>http://localhost:8081</IPaddress>
</ cacheDate>	Users' latest time and date for their status data by their own status update or status liked by their friends.	<cacheDate>Tue, 23 Apr 2019 12:24:12 GMT</cacheDate>
</user>	Use's each posted status	

	data	
</data>		

cache_server.py – a lightweight Python server script only to handle requests from user's message that recording user's latest time and date for their status update for caching purpose.

1.3 Port Numbers

In order to test the system, it needs to run 3 x server.py and 1 x cache_server.py at the same time to simulate the distributed social network system and its cache system on one computer. Then,

cache has port 8080

user1 has port 8081

user2 has port 8082

user3 has port 8083

2. Status update

2.1 Test set up:

- Firefox browser has been used for testing this assignment, other IE browsers might have issues for some functions not working, which have not been fixed.

-Before testing this programme, browser cache should be cleaned.

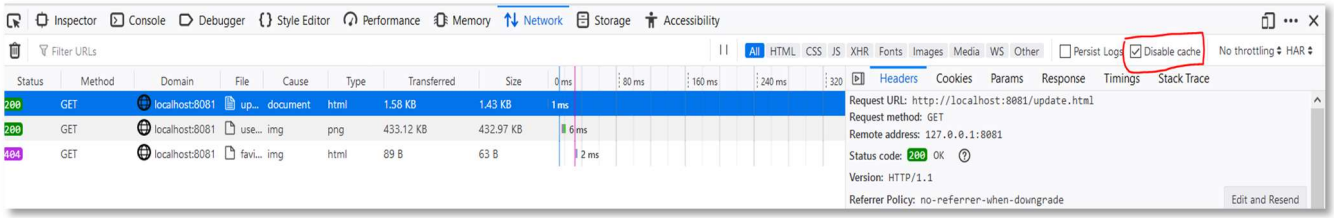
-user1 server has been used for the test of this section, which has port number 8081. User 2 and user3 server will also have the same functions.

-status.xml should be set up as below, <friendlist> and <IPaddress> tag has been pre-set for user1's friend's IP addresses and should not be deleted.

```
<?xml version='1.0' encoding='UTF-8'?>
<data ipAddress="http://localhost:8081" userID="user1">

<friendlist>
<IPaddress>http://localhost:8082</IPaddress>
<IPaddress>http://localhost:8083</IPaddress>
</friendlist></data>
```

-Tick Disable cache box at this stage.



- run server.py in user1 folder and cache_server.py in cache folder now

2.2 Display existing status, user profile picture, friends.html link

2.2.1 Display initial update.html page

-Open http://localhost:8081/update.html now

-The initial update.html is as screenshot below:

2.2.2 Source code description

display_status_in_update_html () and display_friendlist_in_update_html () functions in server_help_functions.py have been used in server.py to render information in update.html file.

2.3 Display post and reset status function

2.3.1 Display reset function

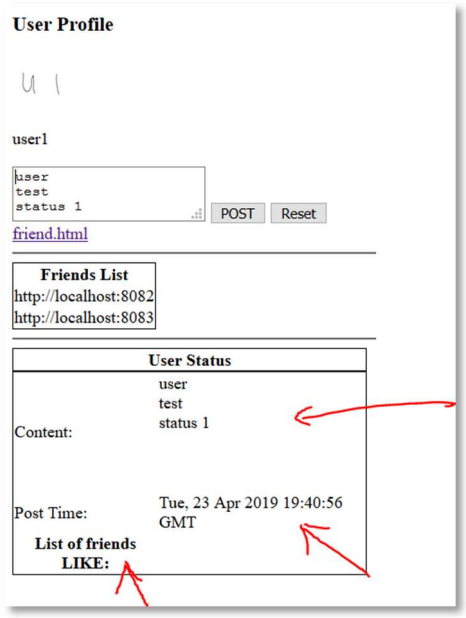
-Input some message in textbox, then click Reset button

-textbox has been reset

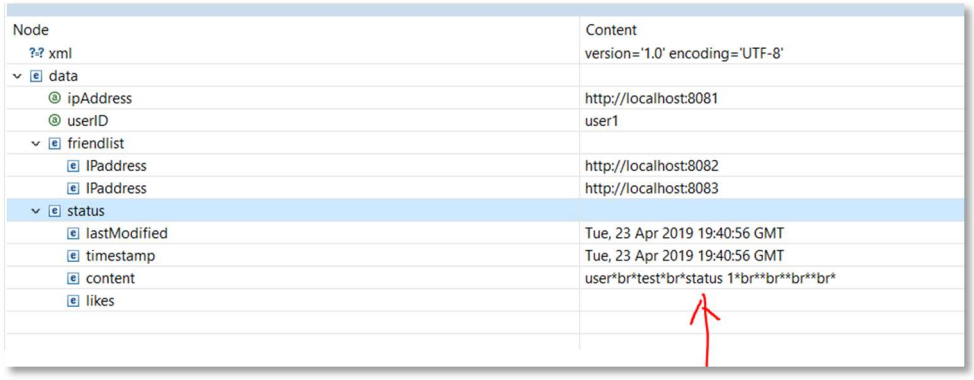
2.3.2 Display POST function

-Input some message in textbox, then click POST button

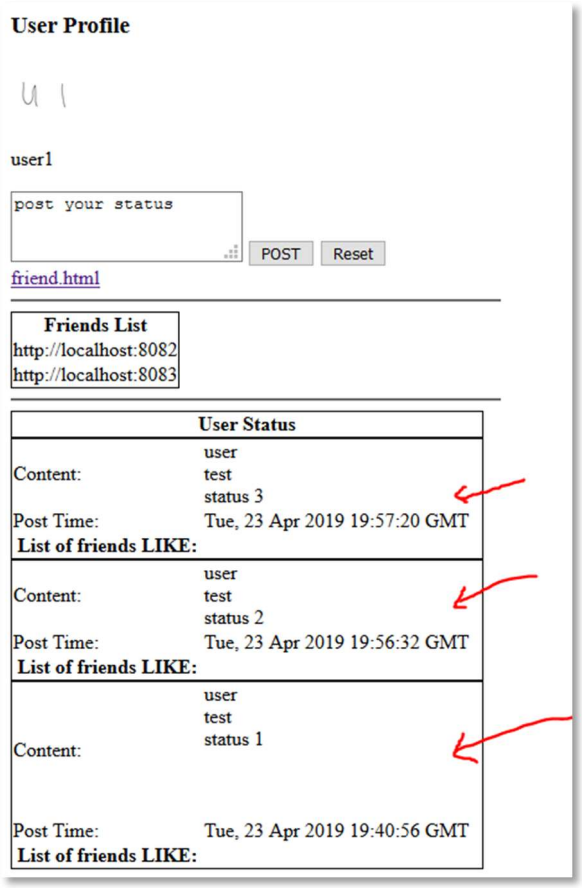
-This user's status has been updated with content, post time and his/her List of friends LIKE, which will be presented in the next section.



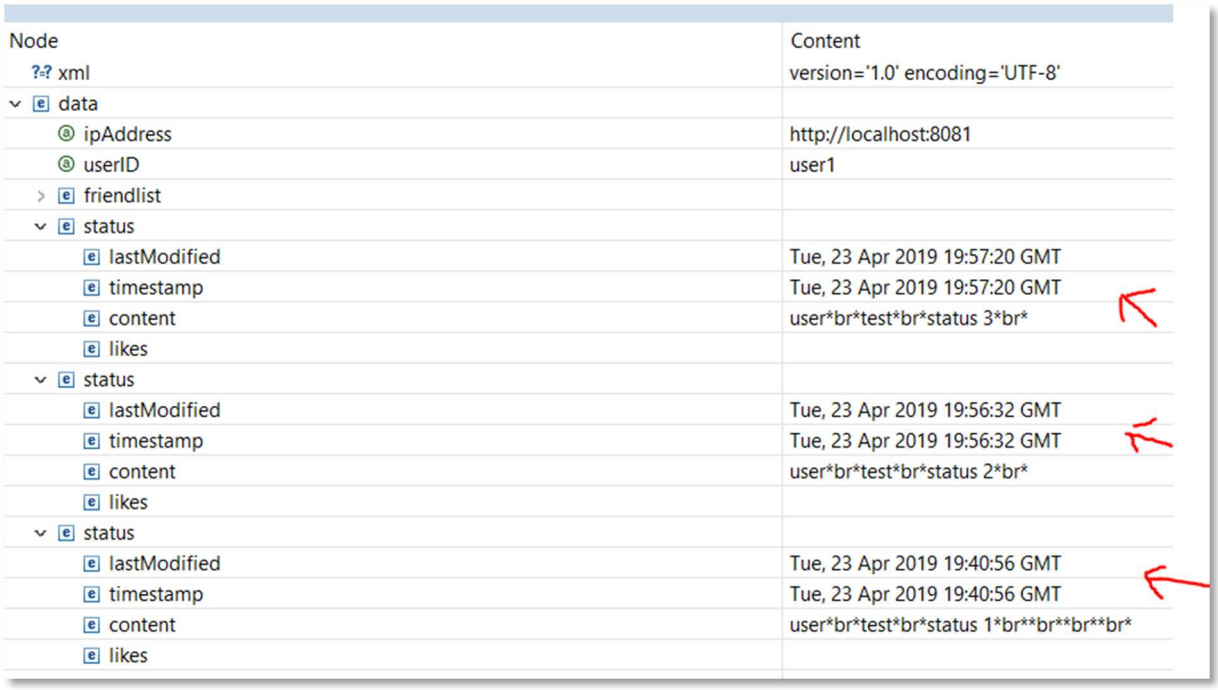
-In status.xml file this information has been appended. Please note the line breaks in textbox will be displayed as "\r\n" from http request, it has been transformed to *br* and then transformed to
 in html page, since
 will be treated at tag in xml file



-If post more status, latest user's status will be put at the top.



- status.xml file has been updated for more status updates as well , its <lastModifed> and <timestamp> tag have the same time and date



-2.3.3 Source code description

add_status(filename,message,serverPort) ,update_status_xml(timestamp,lastmodified,statusContent, serverPort) functions in server_help_functions.py have been used in server.py to render current user's status in update.html file.

add_status(filename,message,serverPort) – filename and message parameters parse the information generated by POST function from browser, which are </timestamp>,</lastmodified>, </statusContent> as a new status record for current user.

update_status_xml(timestamp,lastmodified,statusContent,serverPort) – append new <timestamp>, <lastModified>,<content>,<likes> elements in current user's status.xml file

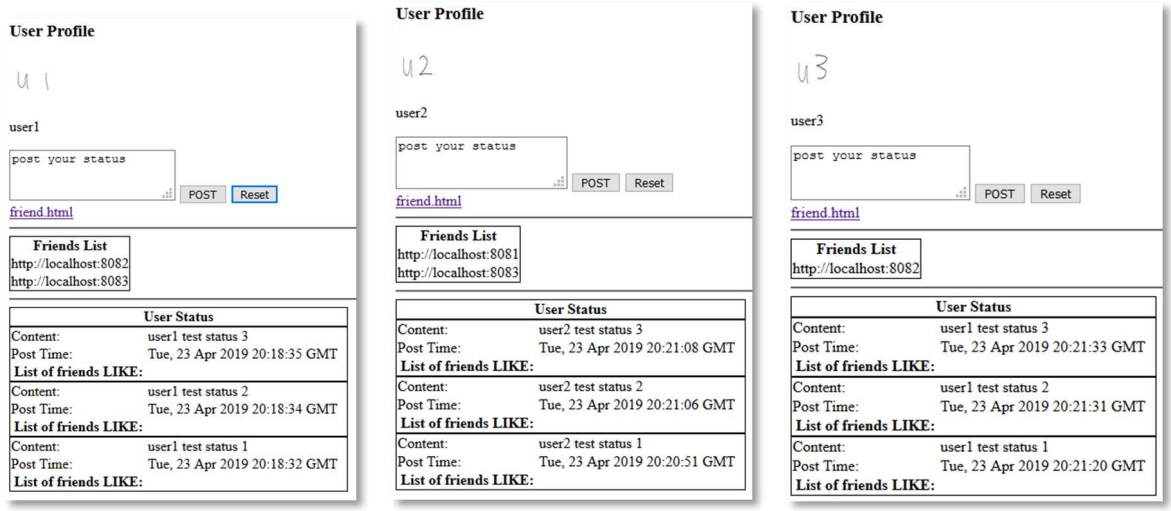
3. Show Friends

3.1 Test set up:

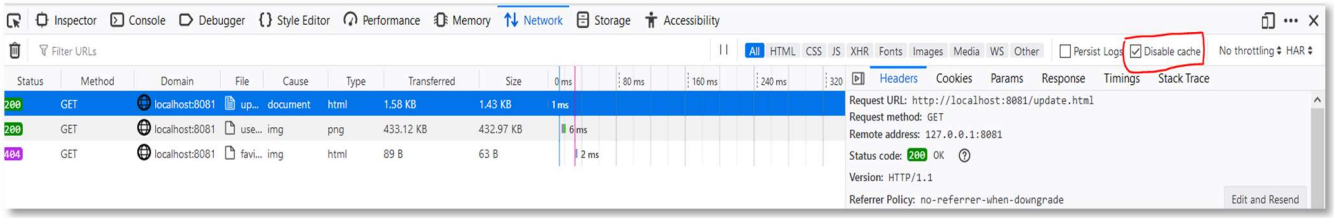
- Firefox browser has been used for testing this assignment, other IE browsers might have issues for some functions not working which have not been fixed.

-Before testing this programme, browser cache should be cleaned.

-user1, user2, user3 servers have been used for the test of this section, which has port number 8081, 8082, 8083. 3 x users' status have been pre-set as below for the test of this section.



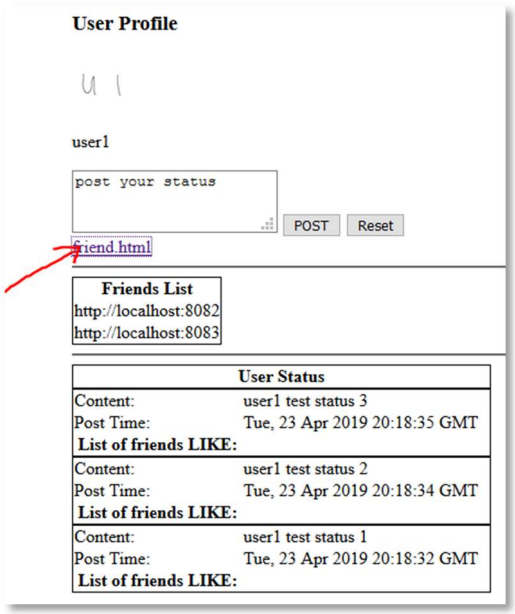
-Tick Disable cache box at this stage.



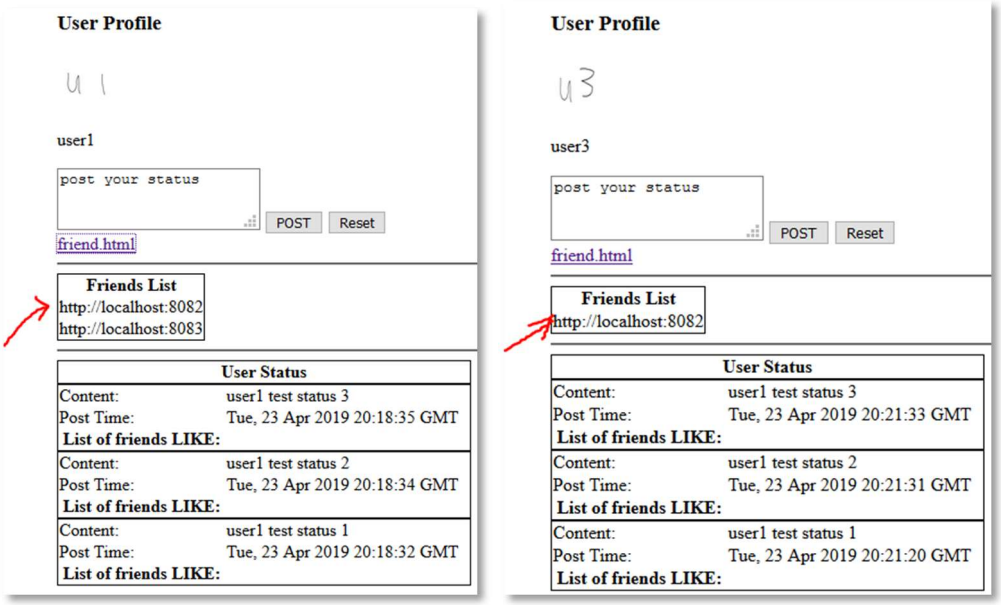
- run 3 x server.py in user1, user2 and user3 folders and cache_server.py in cache folder now

3.2 display user's friends' status, profile picture, like button and number of likes for each status after click friend.html, display The friend's HTTP server will only grant access to the requested object iff the user (HTTP server) requesting the data is in it's own friend list.

- Click friend.html link in update.html of user 1 (http://localhost:8081/update.html)

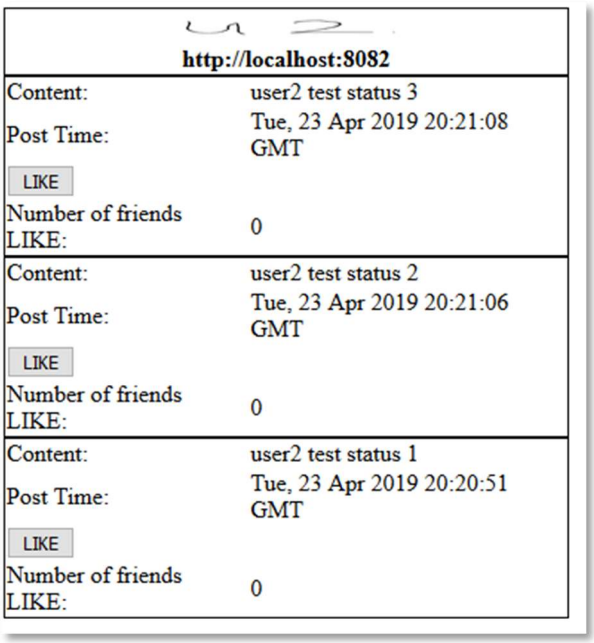


-User1 (http://localhost:8081/update.html) has user 2 and user 3 as friends, however user3 (http://localhost:8083/update.html) only has user2 as friends



.

-Therefore only user2's status could displayed with in friend.html page of user1 (http://localhost:8081/friend.html)

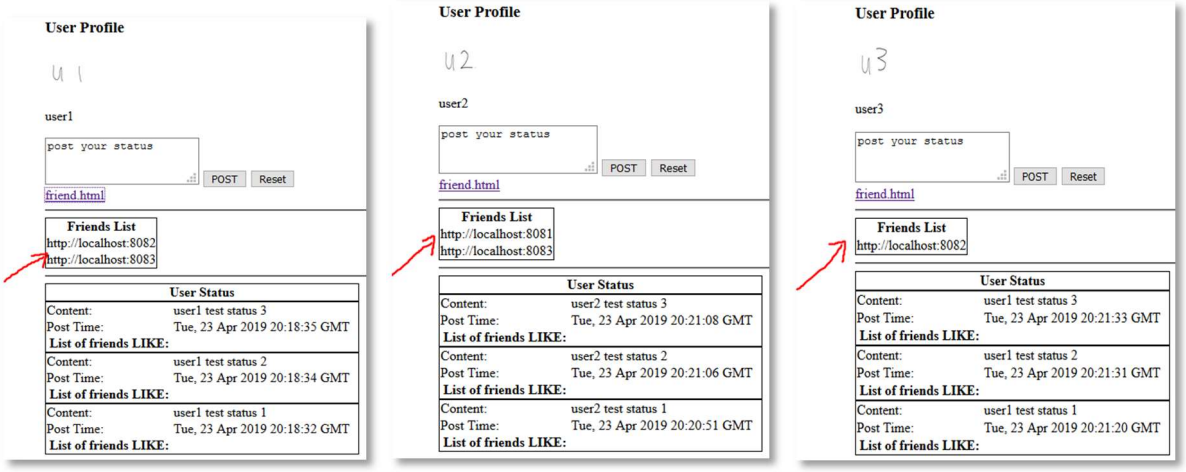


u 2

http://localhost:8082

Content:	user2 test status 3
Post Time:	Tue, 23 Apr 2019 20:21:08 GMT
LIKE	
Number of friends LIKE:	0
Content:	user2 test status 2
Post Time:	Tue, 23 Apr 2019 20:21:06 GMT
LIKE	
Number of friends LIKE:	0
Content:	user2 test status 1
Post Time:	Tue, 23 Apr 2019 20:20:51 GMT
LIKE	
Number of friends LIKE:	0

-Similarly if click firend.html link in update.html of user2 and user3, since user2 has user1 and user3 as friends, user3 has user2 as friends, and user1 has user2 and user3 as friends.



User Profile

u 1

user1

post your status

POST Reset

friend.html

Friends List
http://localhost:8082
http://localhost:8083

User Status

Content:	user1 test status 3
Post Time:	Tue, 23 Apr 2019 20:18:35 GMT
List of friends LIKE:	
Content:	user1 test status 2
Post Time:	Tue, 23 Apr 2019 20:18:34 GMT
List of friends LIKE:	
Content:	user1 test status 1
Post Time:	Tue, 23 Apr 2019 20:18:32 GMT
List of friends LIKE:	

User Profile

u2

user2

post your status

POST Reset

friend.html

Friends List
http://localhost:8081
http://localhost:8083

User Status

Content:	user2 test status 3
Post Time:	Tue, 23 Apr 2019 20:21:08 GMT
List of friends LIKE:	
Content:	user2 test status 2
Post Time:	Tue, 23 Apr 2019 20:21:06 GMT
List of friends LIKE:	
Content:	user2 test status 1
Post Time:	Tue, 23 Apr 2019 20:20:51 GMT
List of friends LIKE:	

User Profile

u3

user3

post your status

POST Reset

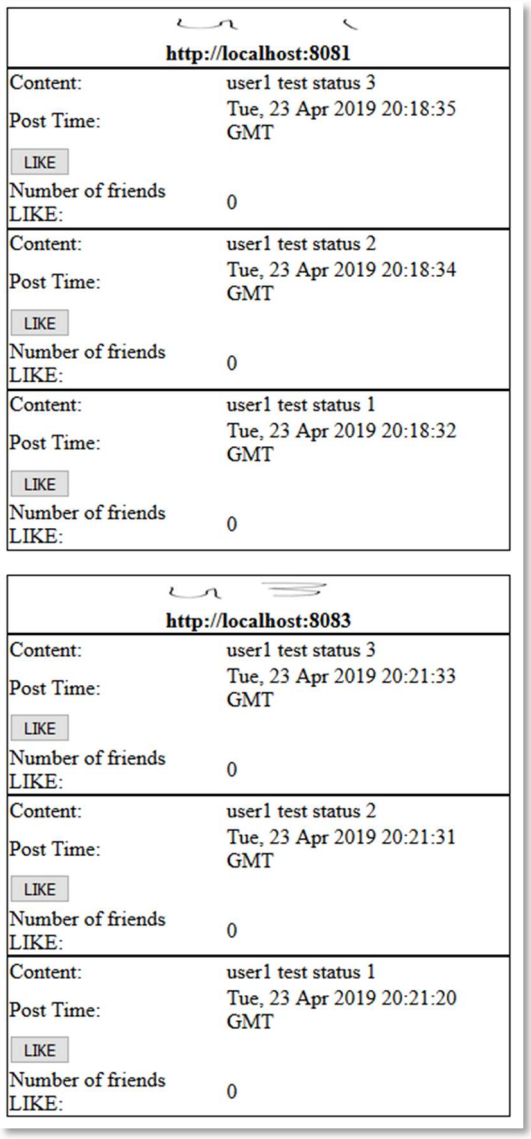
friend.html

Friends List
http://localhost:8082

User Status

Content:	user1 test status 3
Post Time:	Tue, 23 Apr 2019 20:21:33 GMT
List of friends LIKE:	
Content:	user1 test status 2
Post Time:	Tue, 23 Apr 2019 20:21:31 GMT
List of friends LIKE:	
Content:	user1 test status 1
Post Time:	Tue, 23 Apr 2019 20:21:20 GMT
List of friends LIKE:	

-Then user2 (http://localhost:8082/friend.html) could display both status of user1 and user3



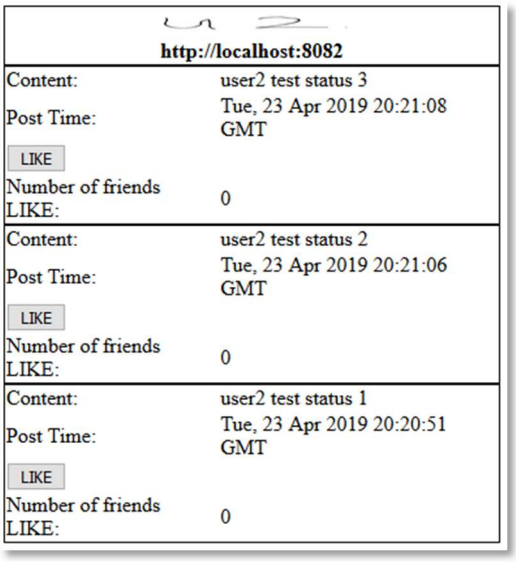
http://localhost:8081

Content:	user1 test status 3
Post Time:	Tue, 23 Apr 2019 20:18:35 GMT
LIKE	
Number of friends LIKE:	0
Content:	user1 test status 2
Post Time:	Tue, 23 Apr 2019 20:18:34 GMT
LIKE	
Number of friends LIKE:	0
Content:	user1 test status 1
Post Time:	Tue, 23 Apr 2019 20:18:32 GMT
LIKE	
Number of friends LIKE:	0

http://localhost:8083

Content:	user1 test status 3
Post Time:	Tue, 23 Apr 2019 20:21:33 GMT
LIKE	
Number of friends LIKE:	0
Content:	user1 test status 2
Post Time:	Tue, 23 Apr 2019 20:21:31 GMT
LIKE	
Number of friends LIKE:	0
Content:	user1 test status 1
Post Time:	Tue, 23 Apr 2019 20:21:20 GMT
LIKE	
Number of friends LIKE:	0

- and user3 (http://localhost:8083/friend.html) could display user2's status.



http://localhost:8082

Content:	user2 test status 3
Post Time:	Tue, 23 Apr 2019 20:21:08 GMT
LIKE	
Number of friends LIKE:	0
Content:	user2 test status 2
Post Time:	Tue, 23 Apr 2019 20:21:06 GMT
LIKE	
Number of friends LIKE:	0
Content:	user2 test status 1
Post Time:	Tue, 23 Apr 2019 20:20:51 GMT
LIKE	
Number of friends LIKE:	0

3.3. Source code description

display_friendStatus_in_friend_html (userIP) function in server_help_functions.py has been used in server.py to render information in friend.html file, userIP parameter will tell friend's server who is visiting them.

hasUserIP (ip_in_friendlist, userIP) function in server_help_functions.py has been used in display_friendStatus_in_friend_html (userIP) function to check if userIP was in his/her friend's friendslist, before rendering their information in friend.html file.

Access_Control_Allow_Origin(message) function server_help_functions.py has been used in to add a Access-Control-Allow-Origin response header ensure user could only response to client iff client request is in user's friend list, message parameter is used to parse client IP address.

4. Like

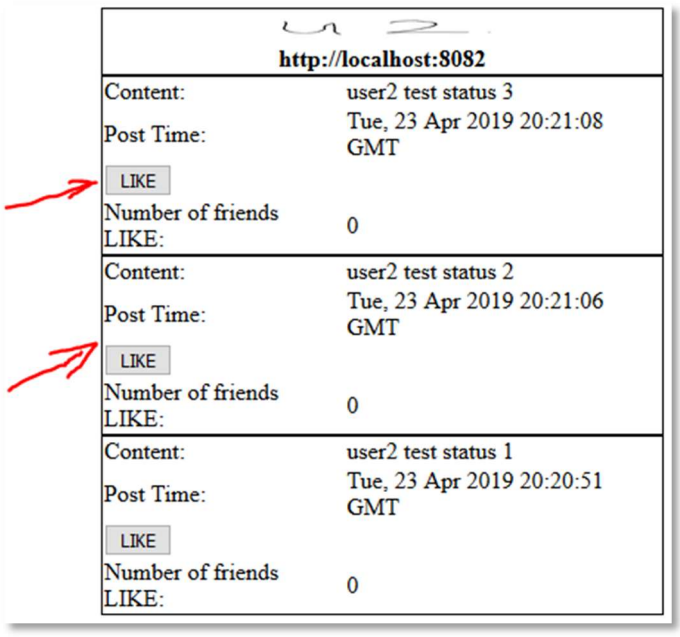
4.1 Test set up:

-Refer to section 3.1, then continue the test from last section.

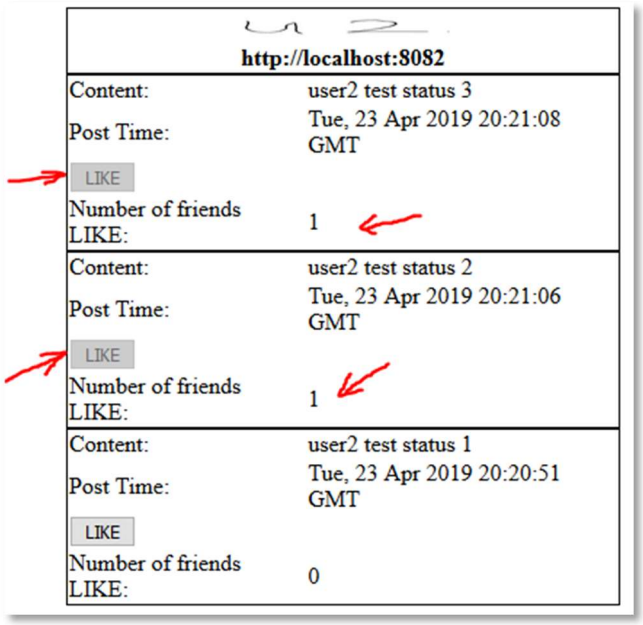
4.2 Display Like function

-Since user2 has user1 and user3 as friends, user3 has user2 as friend, and user1 has user2 and user3 as friends. Then, user2 is a friend for both user1 and user3 will be used for this test.

-In user1's friend.html page (http://localhost:8081/friend.html), user1 click LIKE button for user2 first and second status, and reload page



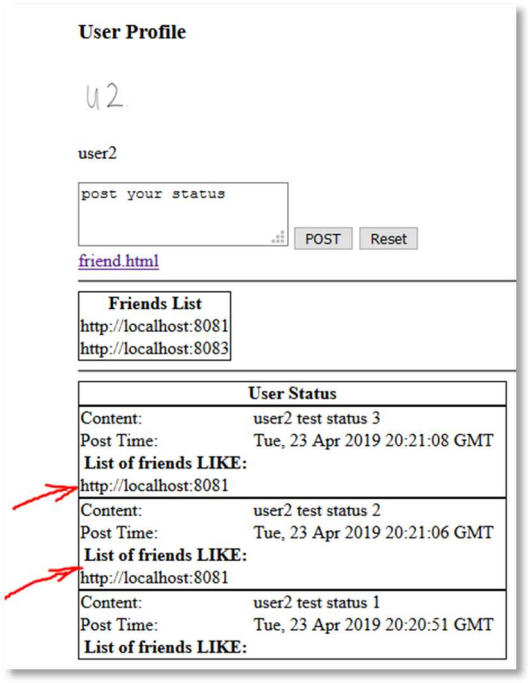
-Then the two LIKE buttons have been turned to grey, Number of Friends have been updated to 1 for both status rows.



http://localhost:8082

Content:	user2 test status 3
Post Time:	Tue, 23 Apr 2019 20:21:08 GMT
LIKE	
Number of friends LIKE:	1
Content:	user2 test status 2
Post Time:	Tue, 23 Apr 2019 20:21:06 GMT
LIKE	
Number of friends LIKE:	1
Content:	user2 test status 1
Post Time:	Tue, 23 Apr 2019 20:20:51 GMT
LIKE	
Number of friends LIKE:	0

-Open user2 update.html (http://localhost:8082/update.html), the user1's IP has been added under List of friends LIKE:



User Profile

user2

post your status

POST Reset

friend.html

Friends List
http://localhost:8081
http://localhost:8083

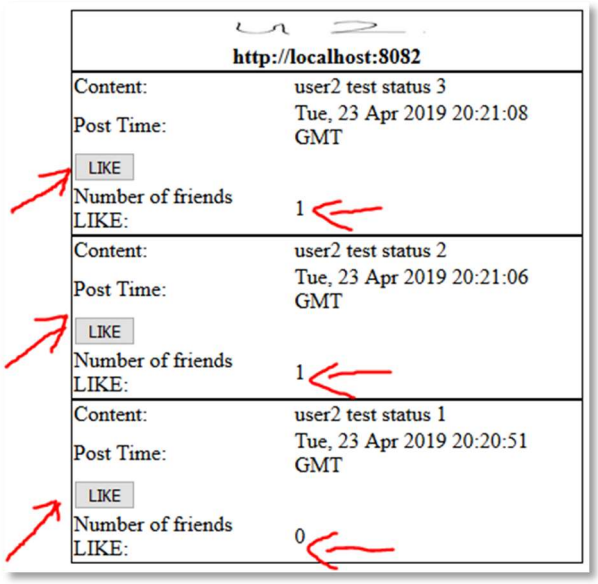
User Status

Content:	user2 test status 3
Post Time:	Tue, 23 Apr 2019 20:21:08 GMT
List of friends LIKE:	
http://localhost:8081	
Content:	user2 test status 2
Post Time:	Tue, 23 Apr 2019 20:21:06 GMT
List of friends LIKE:	
http://localhost:8081	
Content:	user2 test status 1
Post Time:	Tue, 23 Apr 2019 20:20:51 GMT
List of friends LIKE:	

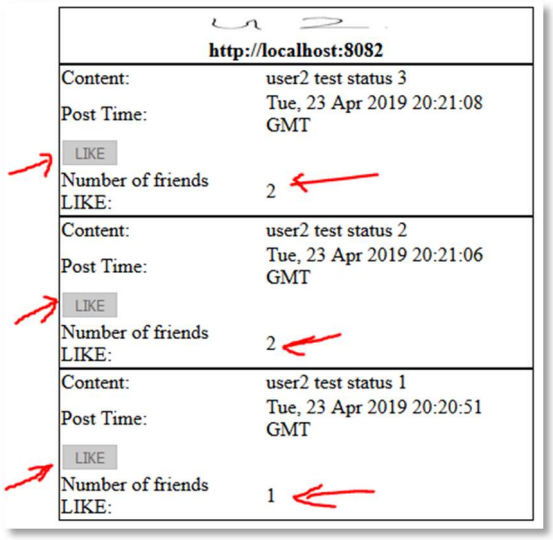
-In its status.xml file, the <lastModified> tag's time stamp has been updated to the time this status row that has been just LIKED, user1's IP address has been added into <likes> tag as well.

Node	Content
?=? xml	version='1.0' encoding='UTF-8'
data	
ipAddress	http://localhost:8082
userID	user2
friendlist	
status	
lastModified	Tue, 23 Apr 2019 21:40:28 GMT
timestamp	Tue, 23 Apr 2019 20:21:08 GMT
content	user2 test status 3*br*
likes	
IPaddress	http://localhost:8081
status	
lastModified	Tue, 23 Apr 2019 21:40:29 GMT
timestamp	Tue, 23 Apr 2019 20:21:06 GMT
content	user2 test status 2*br*
likes	
IPaddress	http://localhost:8081
status	
lastModified	Tue, 23 Apr 2019 20:20:51 GMT
timestamp	Tue, 23 Apr 2019 20:20:51 GMT
content	user2 test status 1*br*
likes	

-Similarly open user3's friend.html page (http://localhost:8083/friend.html), which would be same as last user1's friend.html page, now user3 click LIKE button for user2 three status and reload page.



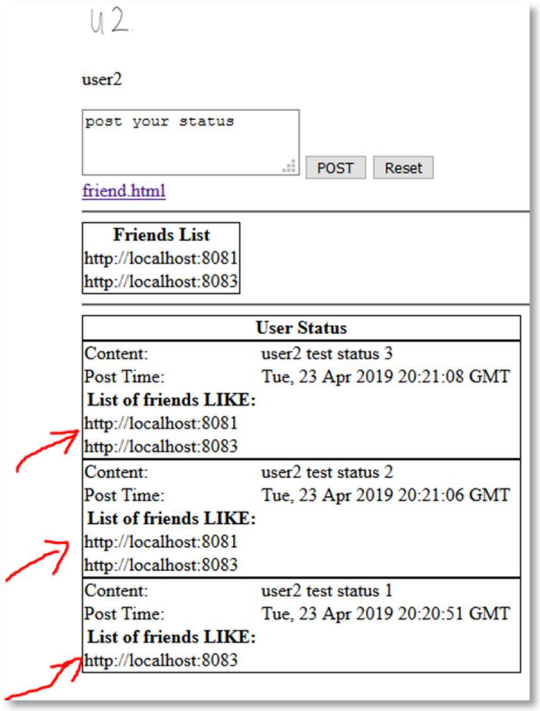
-Then the three LIKE buttons have been turned to grey, Number of Friends have been updated to 2,2, and 1 for three status rows.



http://localhost:8082

Content:	user2 test status 3
Post Time:	Tue, 23 Apr 2019 20:21:08 GMT
LIKE	
Number of friends LIKE:	2
Content:	user2 test status 2
Post Time:	Tue, 23 Apr 2019 20:21:06 GMT
LIKE	
Number of friends LIKE:	2
Content:	user2 test status 1
Post Time:	Tue, 23 Apr 2019 20:20:51 GMT
LIKE	
Number of friends LIKE:	1

-Open user2 update.html (http://localhost:8082/update.html) again, the user3's IP has been added under List of friends LIKE:



user2

post your status

POST Reset

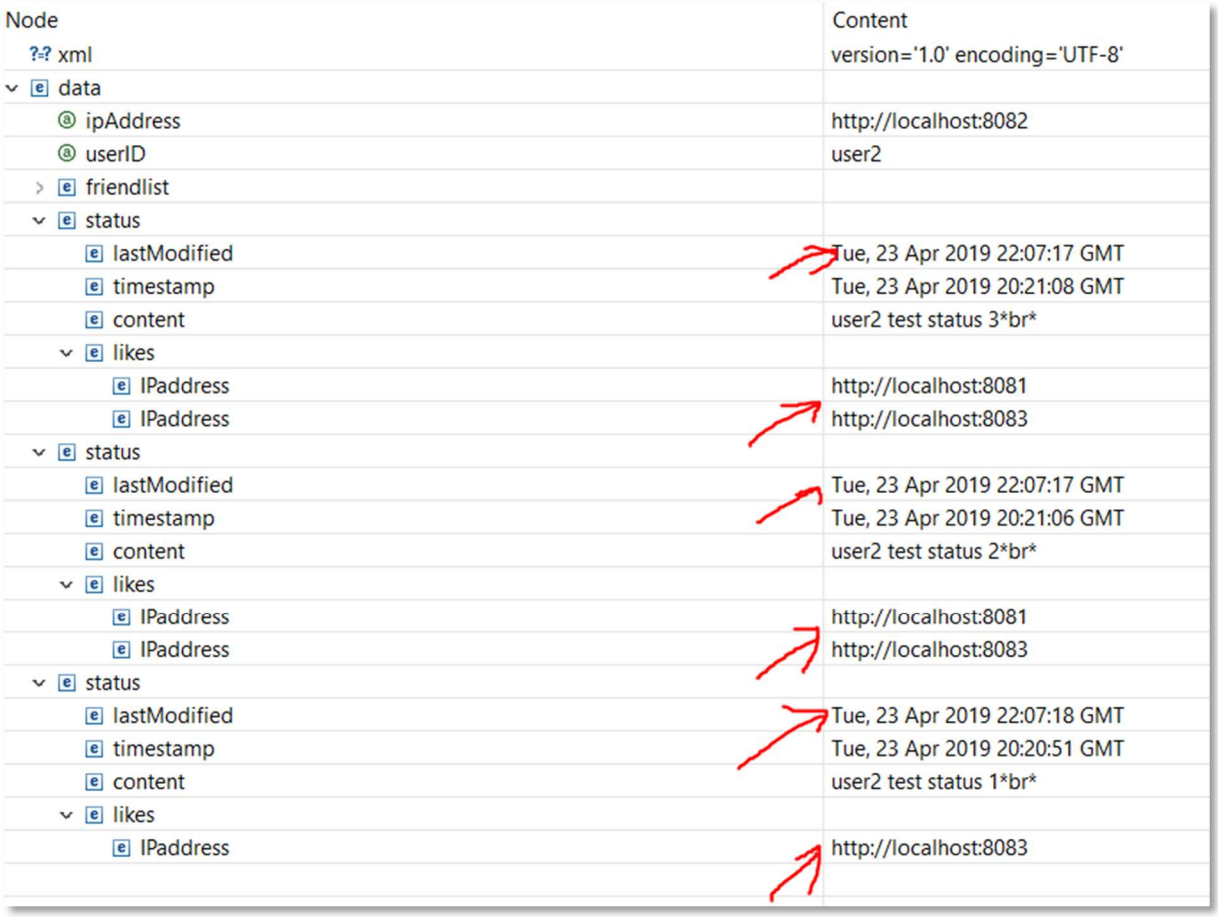
friend.html

Friends List
http://localhost:8081
http://localhost:8083

User Status

Content:	user2 test status 3
Post Time:	Tue, 23 Apr 2019 20:21:08 GMT
List of friends LIKE:	
http://localhost:8081	
http://localhost:8083	
Content:	user2 test status 2
Post Time:	Tue, 23 Apr 2019 20:21:06 GMT
List of friends LIKE:	
http://localhost:8081	
http://localhost:8083	
Content:	user2 test status 1
Post Time:	Tue, 23 Apr 2019 20:20:51 GMT
List of friends LIKE:	
http://localhost:8083	

-In its status.xml file, the <lastModified> tag's time stamp has been updated to the time this status row that has been just LIKED, user3's IP address has been added into <likes> tag as well.



Node	Content
?-? xml	version='1.0' encoding='UTF-8'
data	
ipAddress	http://localhost:8082
userID	user2
friendlist	
status	
lastModified	Tue, 23 Apr 2019 22:07:17 GMT
timestamp	Tue, 23 Apr 2019 20:21:08 GMT
content	user2 test status 3*br*
likes	
IPaddress	http://localhost:8081
IPaddress	http://localhost:8083
status	
lastModified	Tue, 23 Apr 2019 22:07:17 GMT
timestamp	Tue, 23 Apr 2019 20:21:06 GMT
content	user2 test status 2*br*
likes	
IPaddress	http://localhost:8081
IPaddress	http://localhost:8083
status	
lastModified	Tue, 23 Apr 2019 22:07:18 GMT
timestamp	Tue, 23 Apr 2019 20:20:51 GMT
content	user2 test status 1*br*
likes	
IPaddress	http://localhost:8083

4.3. Source code description

post_like(filename,serverPort)) function in server_help_functions.py have been used in server.py to post 'like' message to friend's server.

add_like(filename, serverPort) - filename parameters parse the information generated by POST function from friend's server who has clicked 'like' button, which are userIP and the location of status that has been 'liked' by current user.

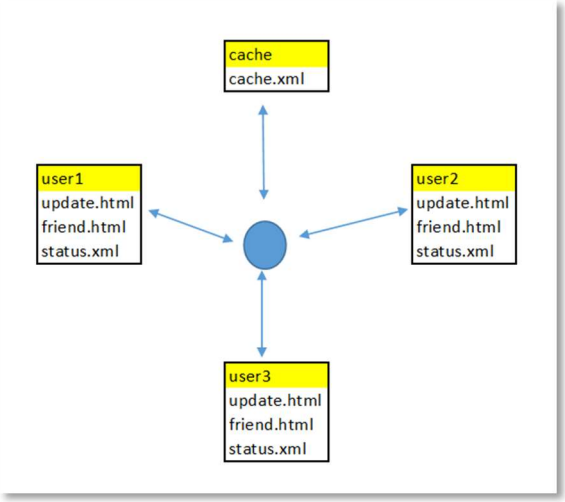
update_status_like_xml(userIP,StatusNumber) – append current user IP to <likes> elements at location of status that has been 'liked' by current user and update < lastModified> time elements for that status in friend.html file

hasliked(ip_in_likes,userIP) function in server_help_functions.py has been used to guard unique IP addresses to be appended, since the cached page in browser would not be refreshed immediately which return a grey 'liked' button, this could make user send information more than twice.

5. System Design

5.1 System conceptual framework

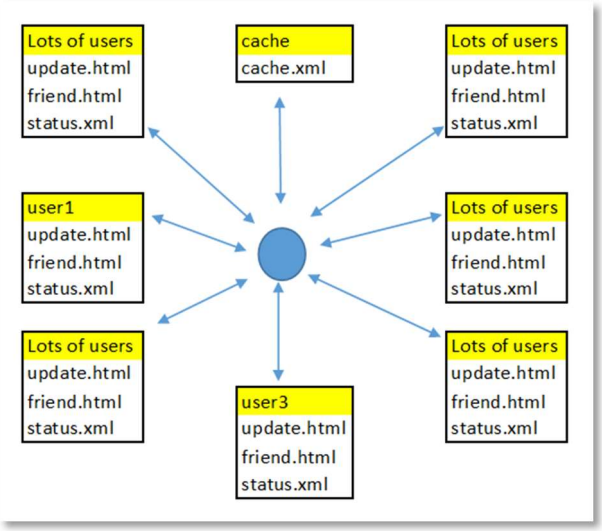
-The conceptual framework is as shown below, an additional cache server has been added in this network, which is used to record all users' last time and date of status update, the cache server will be notified when a user adds a new status in their update.html page, or in its friend.html, when he LIKED his friend's status or his friend's status has been updated by his friend or LIKED by another user in his friend's list …….



-When displaying friend.html page, user needs to iterate through each friend's status.xml file to render the friend.html page, however, it would be very inefficient if the networking growing very big for user to interact a lot of friend's status.xml file as per image below.

As per caching purpose, in a user's update.html page the Last-Modified response header could be computed in current user's <lastModified> data in status.xml. In friend.html page, its Last-Modified response header could be computed by this user iterates all his friend's <lastModified> data in status.xml, which would be very slow if user has a very big friend's network.

Therefore, a cache server has been designed and added to store all user's last update time and date in cache.xml file once a user in this network has "LIKE" or "Content" of his status been updated, then current user only need to request cache server one time in order to compute the last update time and date from its friend's status, which could request server to response 304 Not Modified response



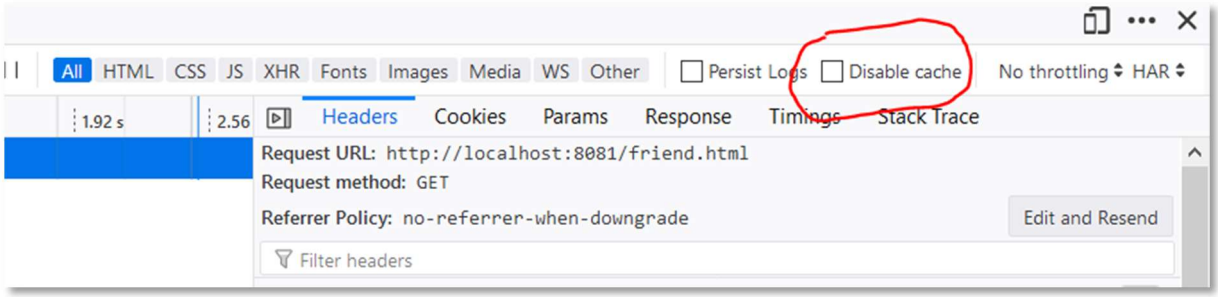
5.2 Test cache function

5.2.1 Test set up:

-cache.xml has been used to store all user's last update time and date in <cachDate> tag.

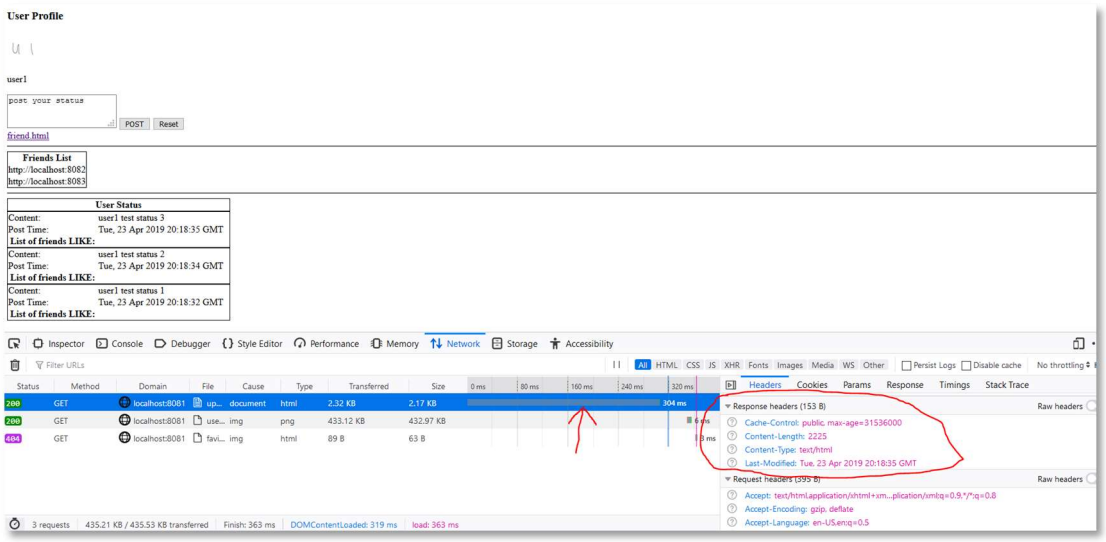
Node	Content
?=? xml	version='1.0' encoding='UTF-8'
data	
ipAddress	http://localhost:8080
userID	cache
user	
userID	user1
IPaddress	http://localhost:8081
cacheDate	Tue, 23 Apr 2019 20:18:35 GMT
user	
userID	user2
IPaddress	http://localhost:8082
cacheDate	Tue, 23 Apr 2019 22:07:18 GMT
user	
userID	user3
IPaddress	http://localhost:8083
cacheDate	Tue, 23 Apr 2019 20:21:33 GMT

-Turn off Disable cache box in browser now, then continue the test from last section.

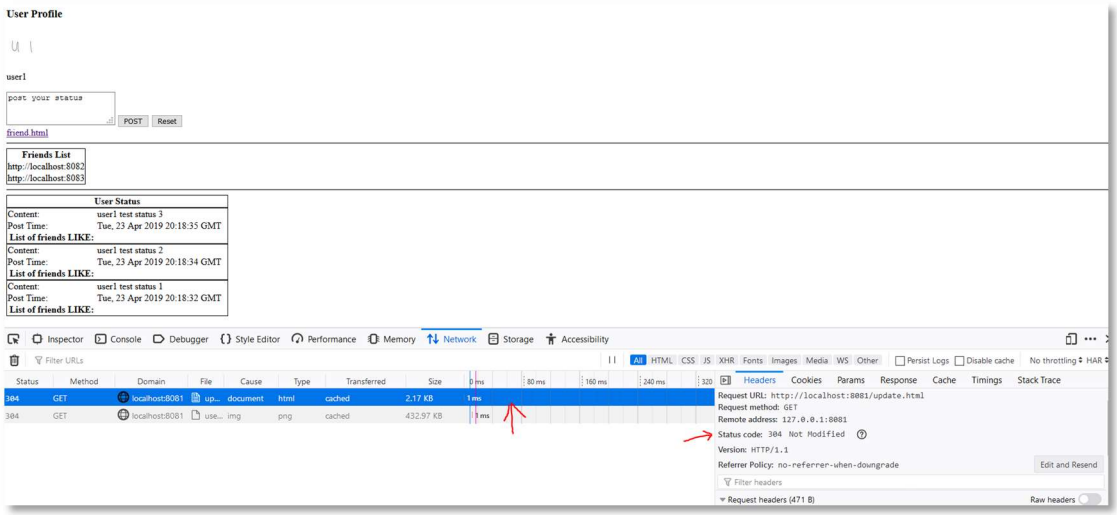


5.2.2 Test update.html page

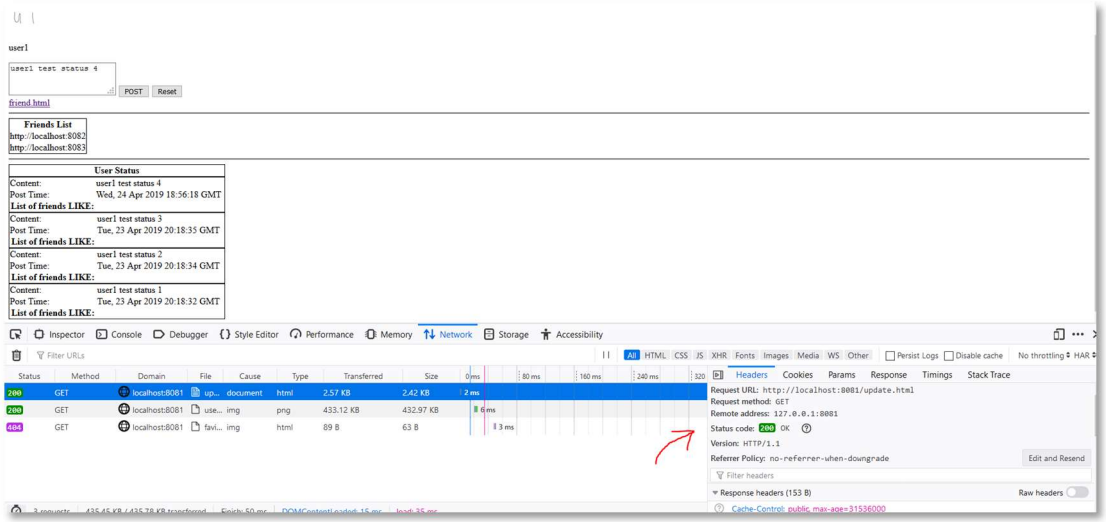
-clean browser cache first, then run 3 x server.py in user1, user2 and user3 folders and cache_server.py in cache folder, open user1 update.html page http://localhost:8081/update.html ,The response time and header as below.



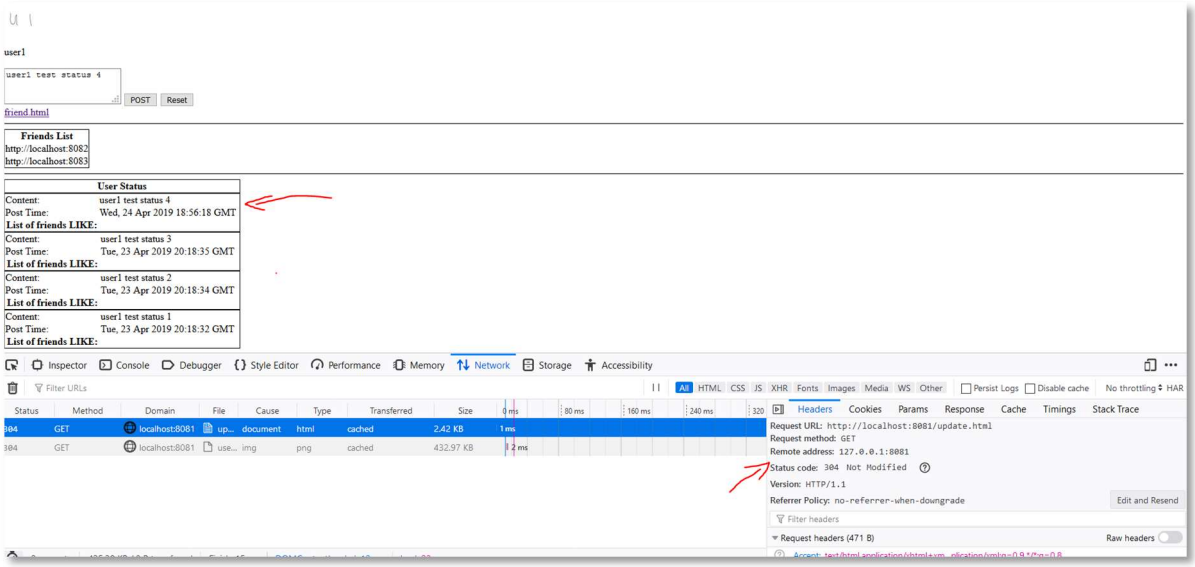
-Reload this page, 304 code has been received, response time as below.



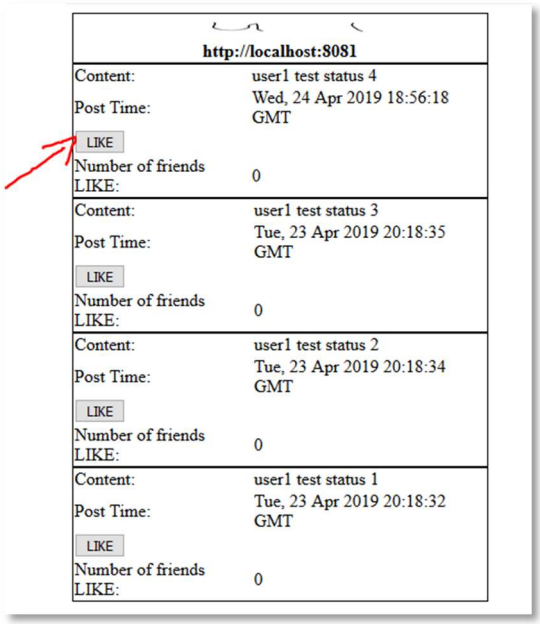
-If post a new status from textbox, page has been updated, 200 code has been received

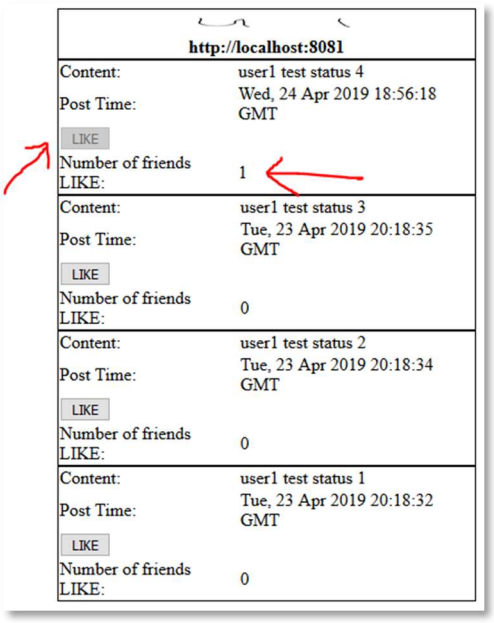


-Reload this page again, 304 code has been received

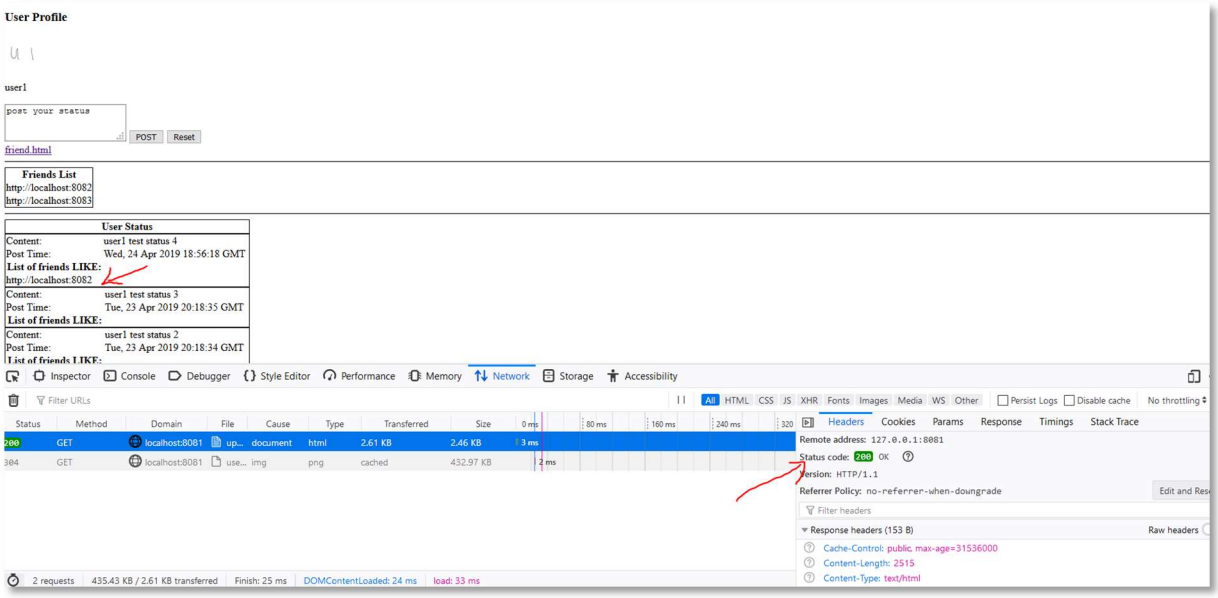


-Now user2 use friend.html page http://localhost:8082/friend.html to like user1's latest status.

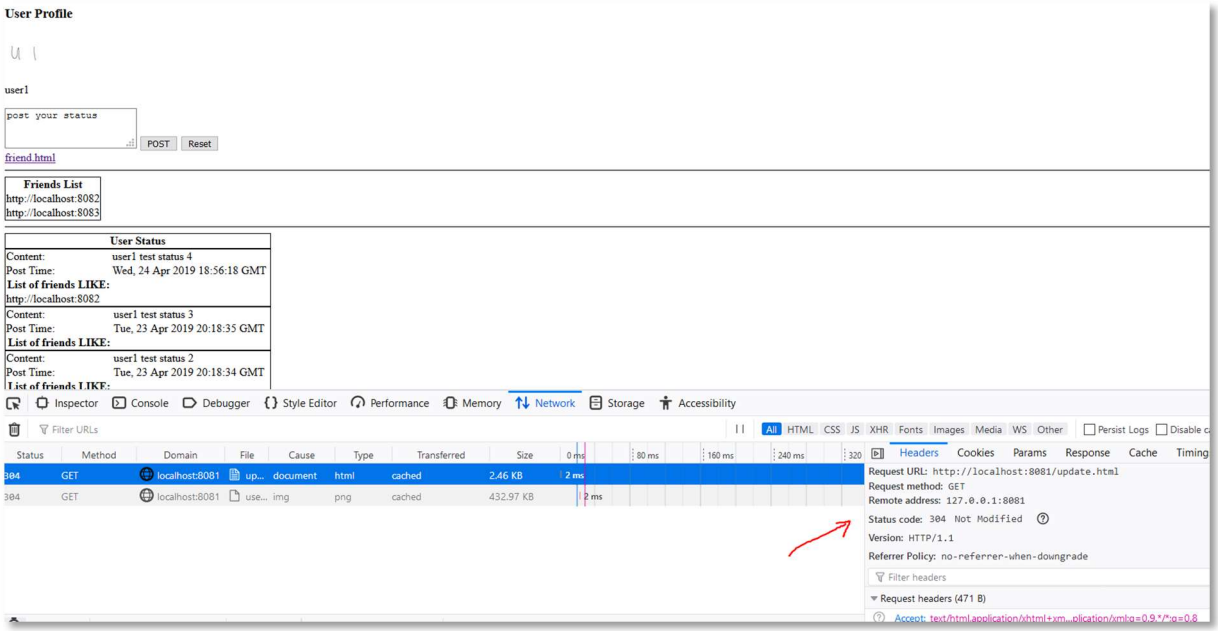




-Reload user1 update.html page http://localhost:8081/update.html , 200 status code received.

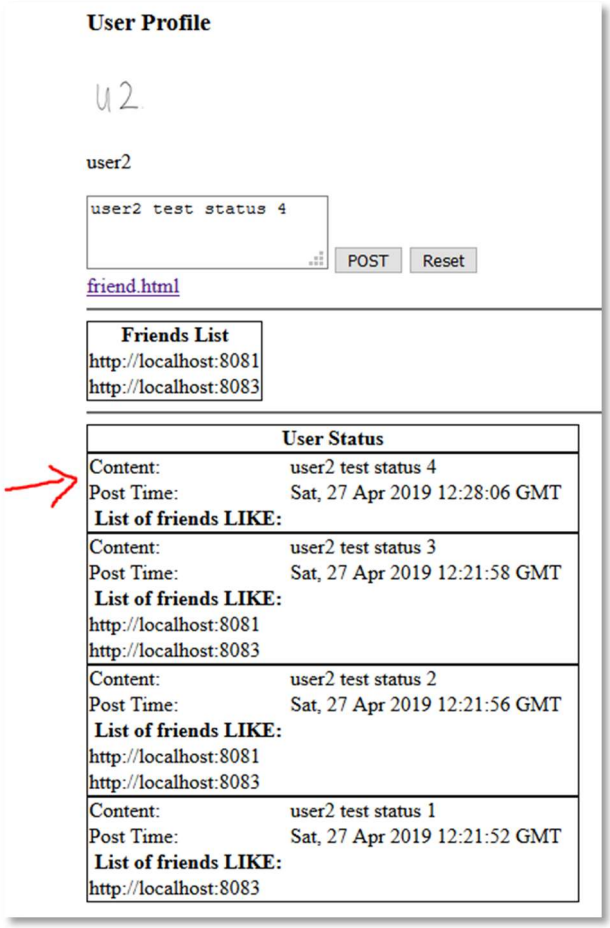


-Reload this page again, 304 code has been received.

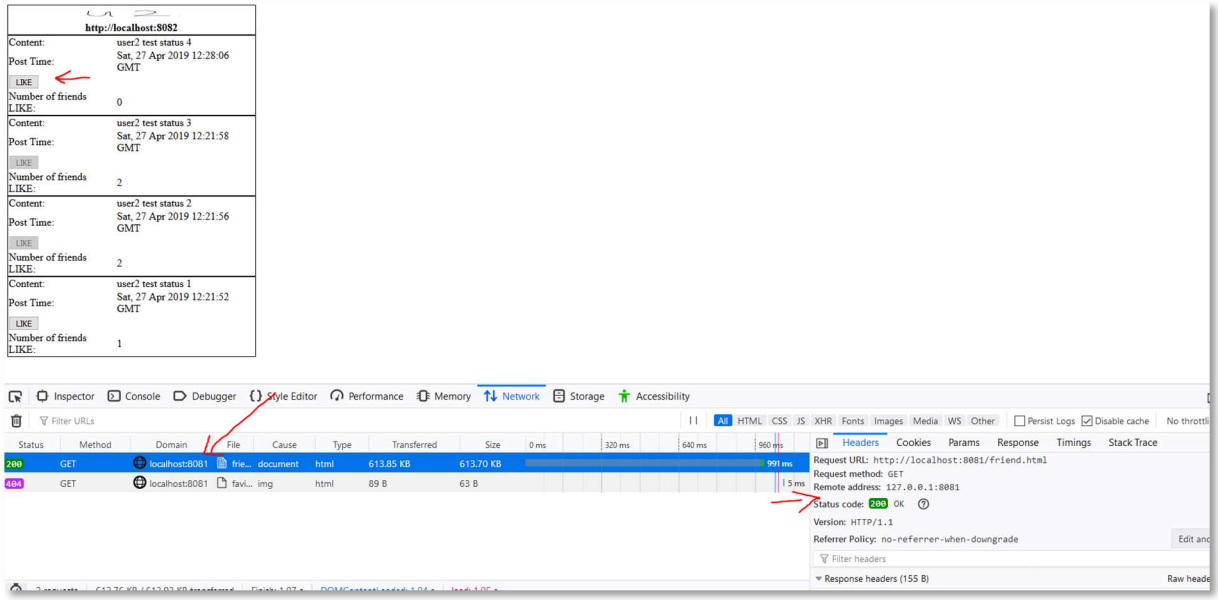


5.2.2 Test friend.html page

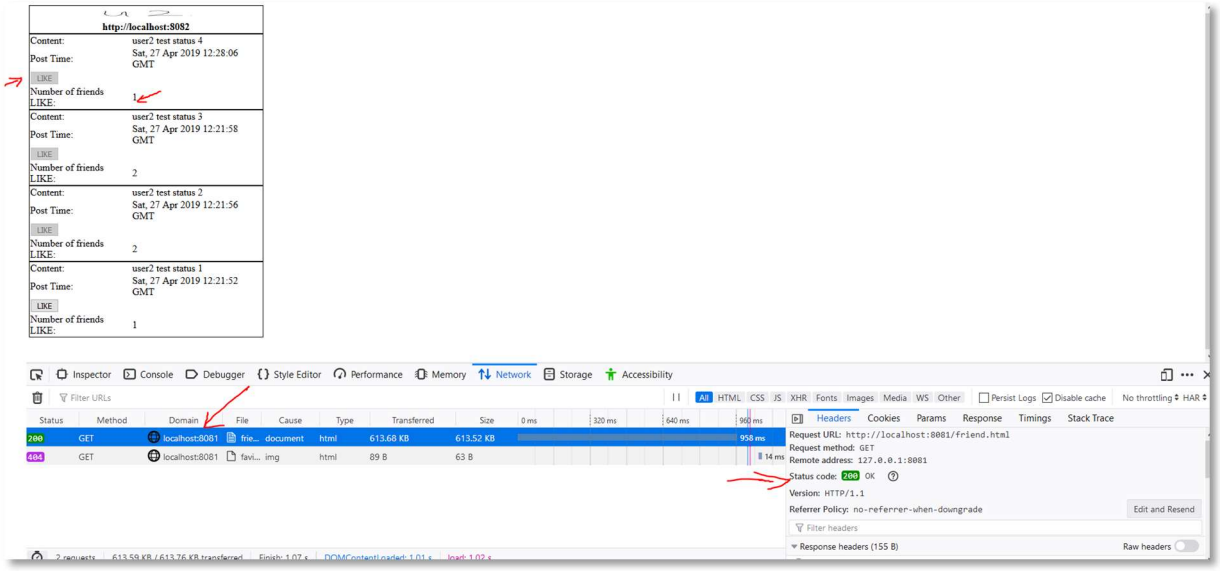
-clean browser cache first, then run 3 x server.py in user1, user2 and user3 folders and cache_server.py in cache folder, since user2 is friend of both user1 and user3, I will use user2 for test. Open user2 update.html page http://localhost:8082/update.html , and add a new status for test.



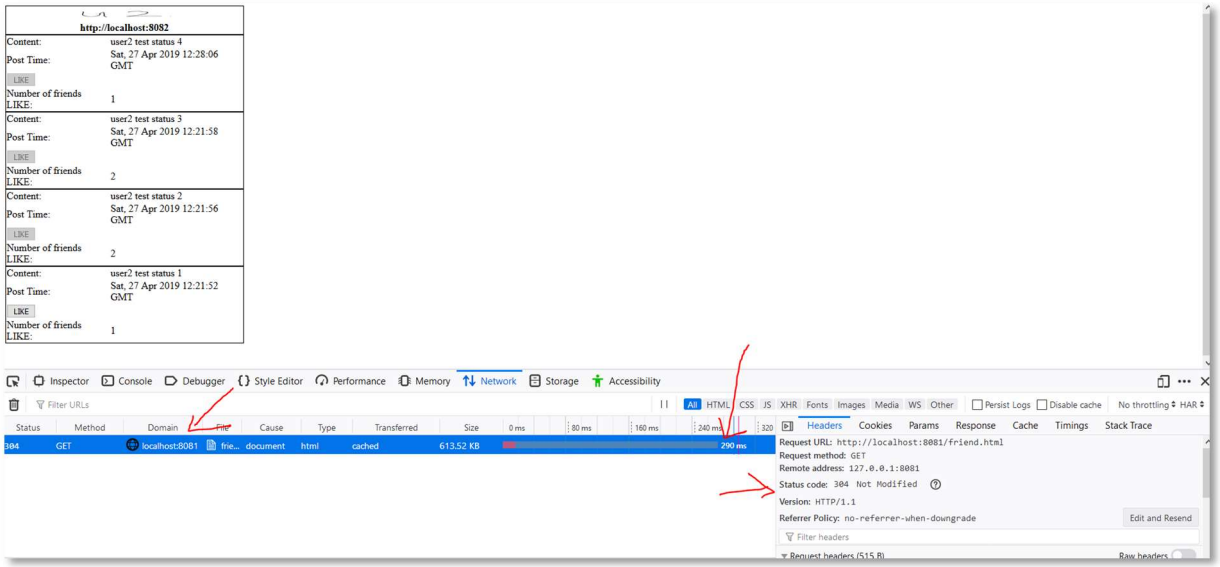
-Open both user1 friend.html page http://localhost:8081/friend.html and user3 friend.html page http://localhost:8083/friend.html. If user1 like user2's latest status.



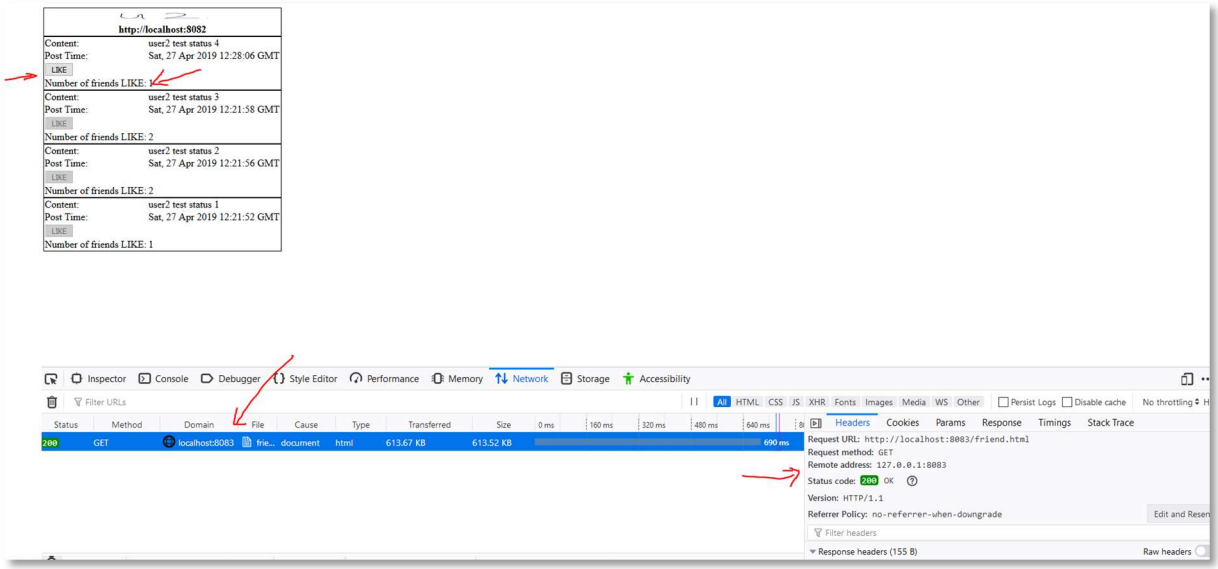
-Reload http://localhost:8081/friend.html page, 200 code has been received.



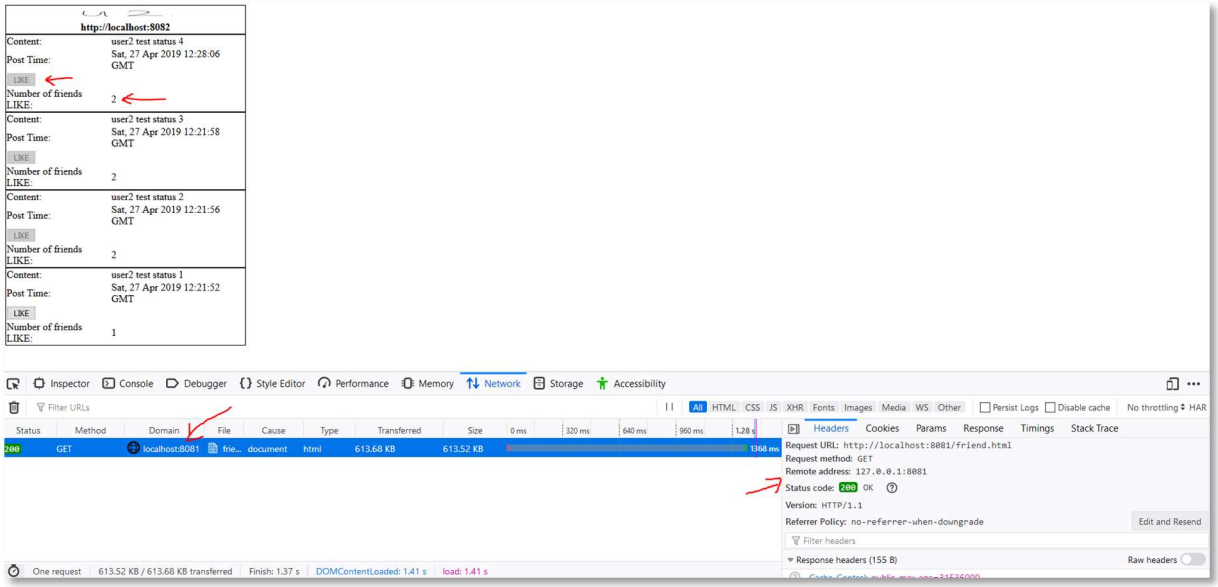
-Reload this again, now 304 code has been received, response time has been reduced to a constant 230ms, since it will send a request to cache_sever.py for the last modified date of users in this friend.html page.



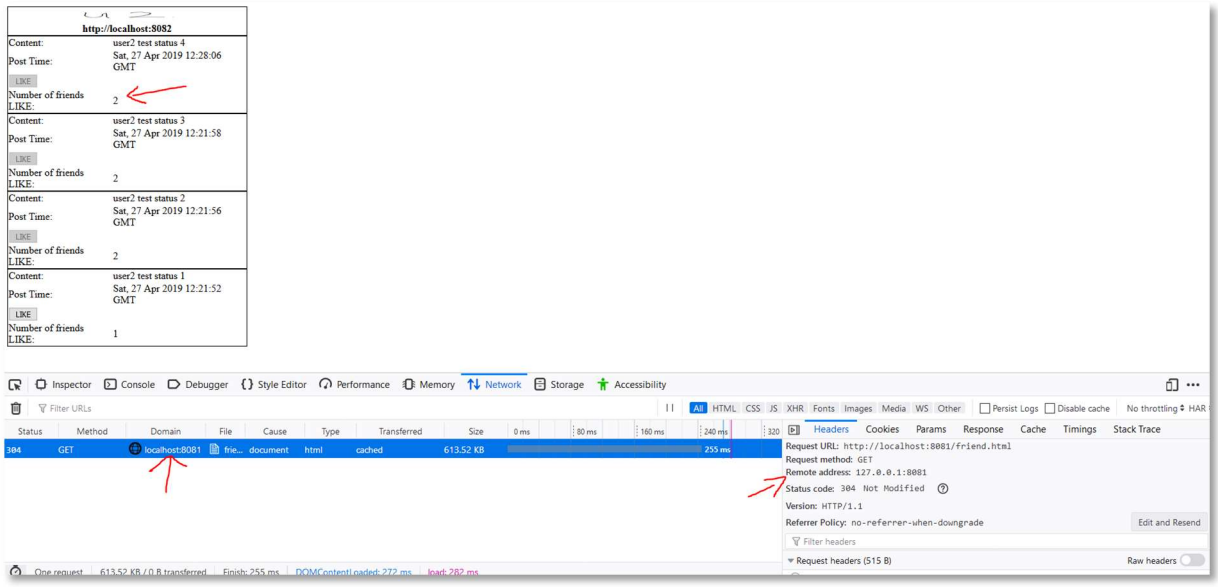
-Now, we switch to user3 friend.html and reload page http://localhost:8083/friend.html. If user3 like user2's latest status.



-The, we switch back to user1 friend.html page http://localhost:8081/friend.html. This page has been updated and 200 code has been received.

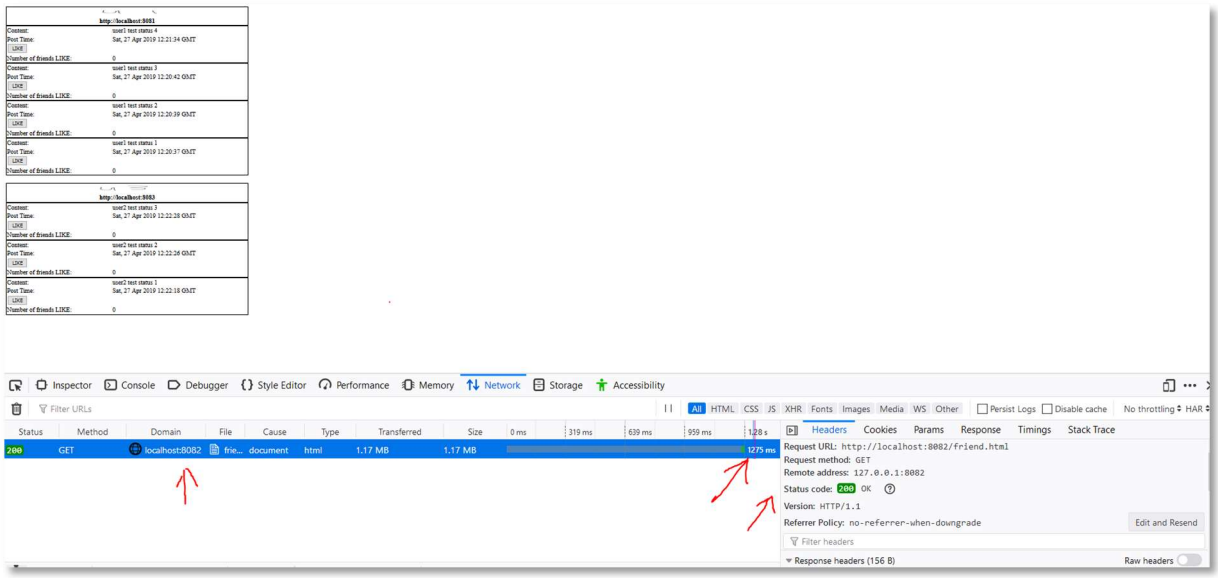


-Reload this page again, it has been cached again, 304 code has been received, since there is no updates in this page.

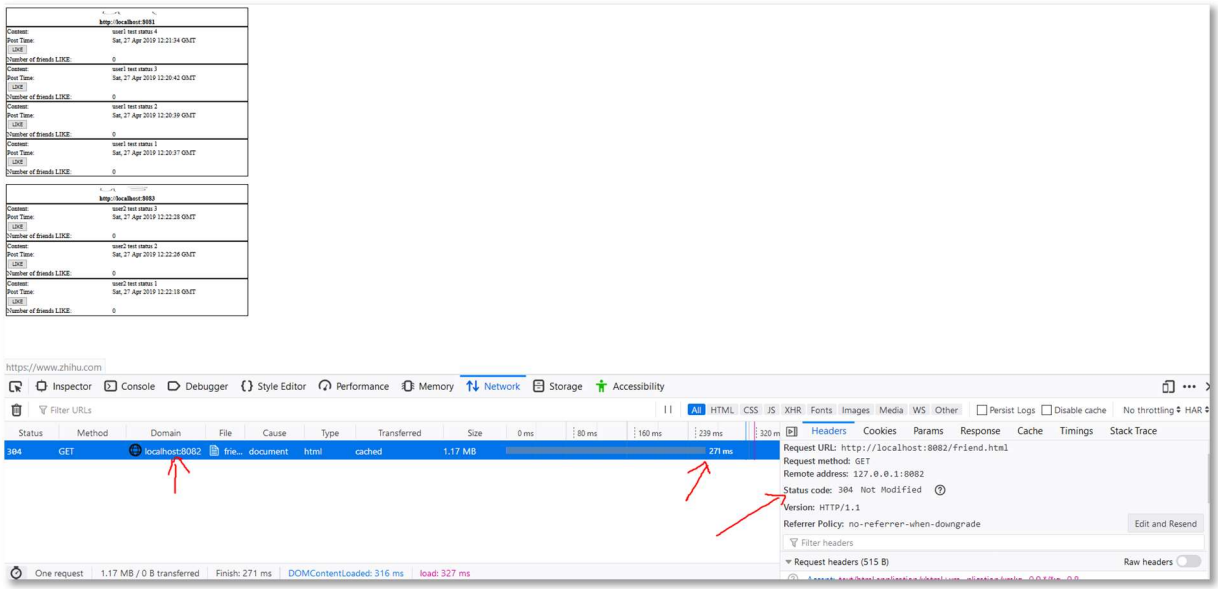


5.2.3 Test friend.html page response time with more than one user status.

-Now we open user2 friend.html page http://localhost:8082/friend.html , which has two users' status displayed. The response time is about 1275ms, since it has iterated two friends' servers.



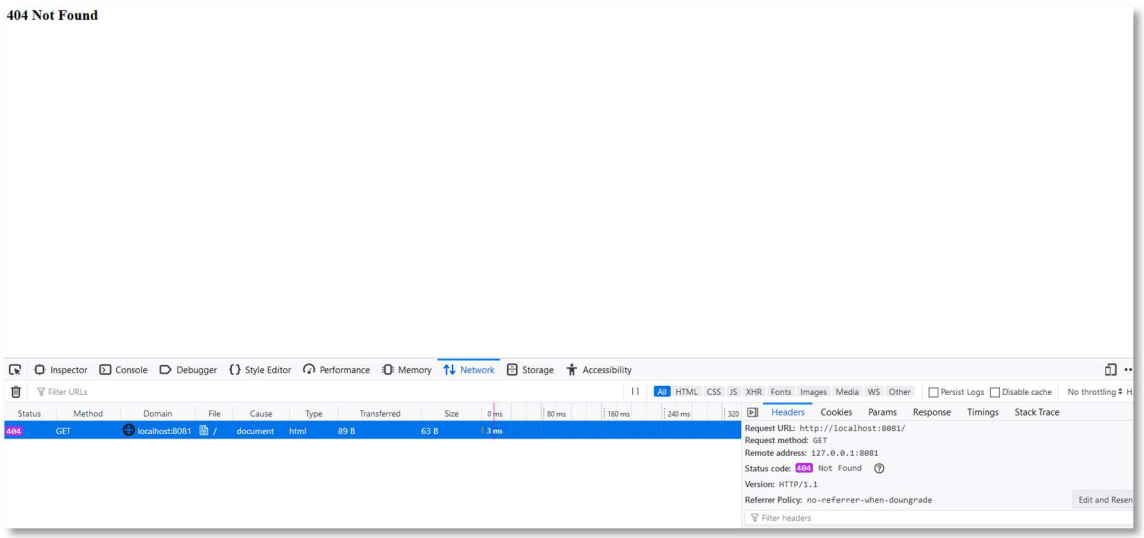
-Reload this page, 304 code has been received, the response time has been reduced to a constant 230-250ms, since it only requests a cache server to check last modified time and date, if the friend's size growing and there is no updates in friends' status , the response time will be remain around 230ms.



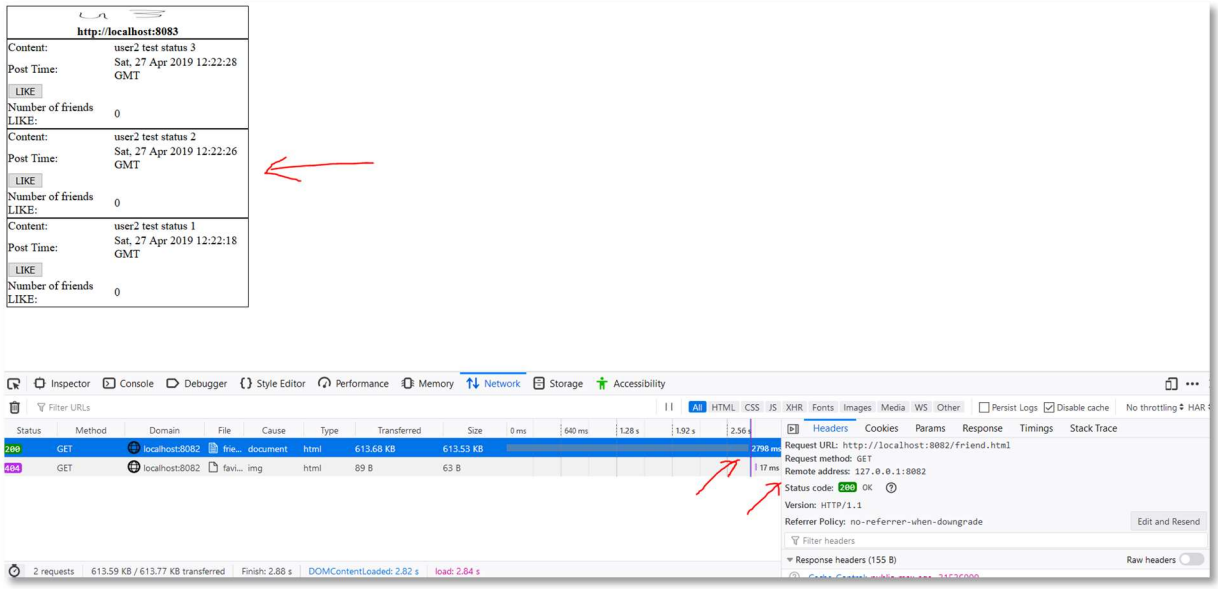
5.3 System decentralised and scalable

5.3.1 System decentralised

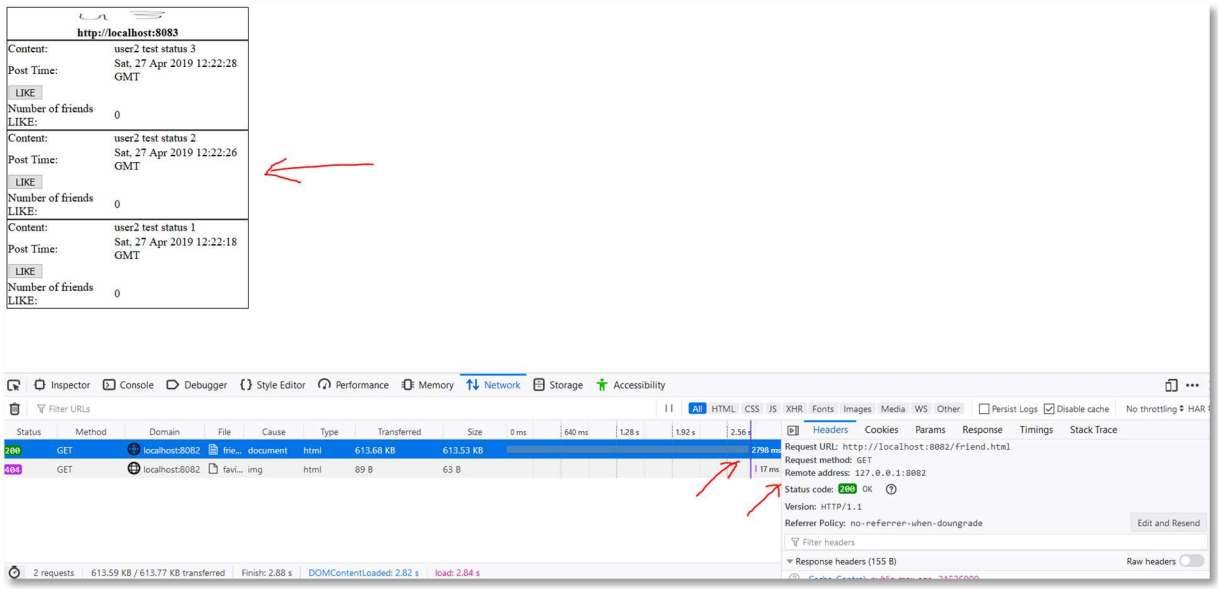
-if we open a non-existing link under user1 server of port 8081, http://localhost:8081/ , 404 Not Found page has been returned.



-Now if we terminate user1 server of port 8081, and open http://localhost:8082/friend.html , since user1 is off, only user3 status has been displayed. It takes long time to check if user1 was off.

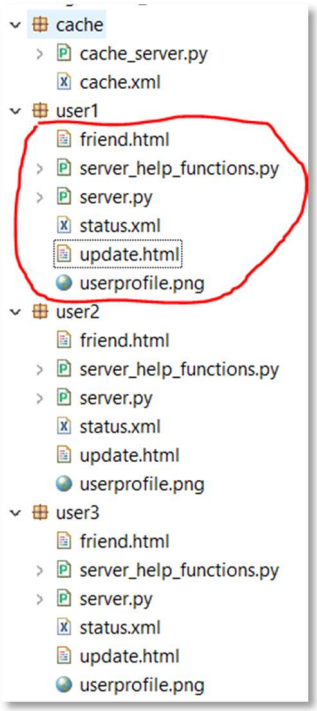


- If turn cache server off now, all other friend.html pages will not perform cache function. The response time will be very slow, since it needs to request cache server first that make the delay.



5.3.1 System scalable.

If more users have been added into this distributed network, it only needs simply to copy block of files and change the server port number in server.py and username in update.html.

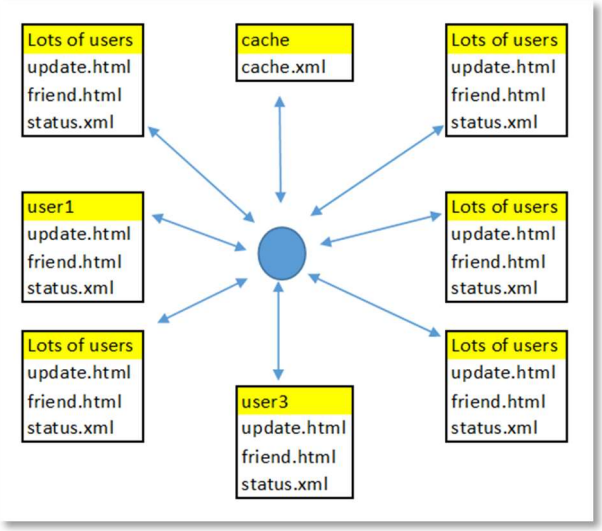



```
<section id = "user">
    <h3>User Profile</h3>
        <img src="userprofile.PNG" alt="user1 profile picture" width="50" height="50"  />
        <p id = "userName">user1</p>
        <form action="new-status" method="post" enctype="multipart/form-data" onsubmit="return submitForm(this);" >
            <textarea name="description" maxlength="200">post your status</textarea>
            <button type="submit">POST</button>
            <button type="reset">Reset</button>
        </form>
        <a href="friend.html">friend.html</a>
        <hr>
</section>

<section id = "friendsList"></section>
<hr>
<section id = "status"></section>
```

6. Further Thoughts

-In this distributed network, the cache server is very crucial, the whole response time will be very slow if this server is down. Therefore, replication of cache server is necessary, just in case if one of them is down, user's request could be redirect to other check server to check last modified date.



-

-Since friend.html page iterates through all user's friend servers, suppose friend1 and friend5 were very popular and their status has been liked frequently, subject to the rest of his friends are very quiet as per image below, then this friend.html page still has to redo iteration of all his user's friends' servers. Therefore, the cache server will not run effectively. A better data structure should be designed that only cache the partition of friend's server data that has not been updated, such as friend2-friend4 and friend6-firend11 as per image below.