
NEW YORK UNIVERSITY
TANDON SCHOOL OF ENGINEERING
CS 6083: PRINCIPLES OF DATABASE SYSTEMS

PROJECT 2

REPORT

Name: Tianyi Zhao

NYU ID: N11142282

netID: tz1330

Date: December 15 2019

I Introduction

This project aims to build a website that allows people to communicate in their neighborhoods. It is based on a user login system in PHP and MySQL. Google Map API is used to specify and display locations. The website provides functions that help people to choose their neighbors and to make friends. People can also send messages to post news and to discuss hobbies and interests.

II Entity-Relational Model

Fig.1 is the ER model of backend database.

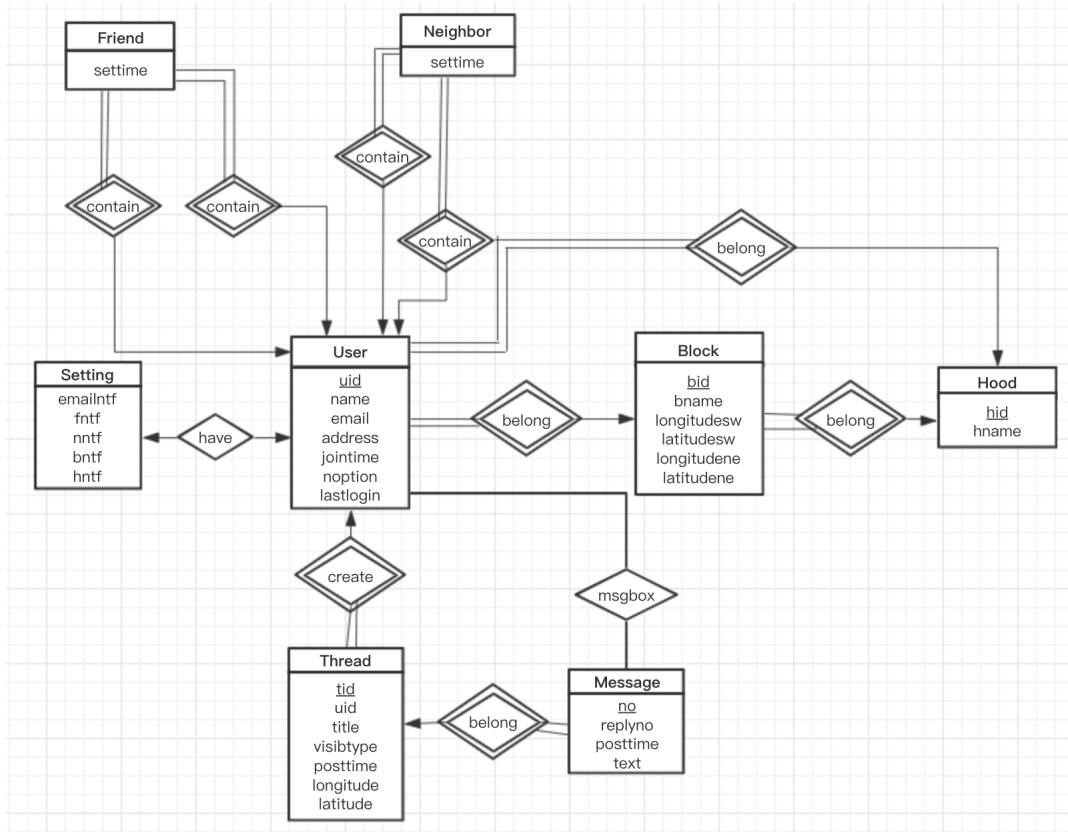


Figure 1: Entity-relational model.

III Relational Schema

The following relational schema is a translation from the ER model. The underlined attributes are primary keys.

- **Hood** (hid, hname)

- **Block** (bid, bname, longitudesw, latitudesw, longitudene, latitudene, hid)
hid references hid in Hood.
- **User** (uid, name, email, address, password, profile, bid, hid, jointime, lastlogin)
bid references bid in Block. hid references hid in Hood.
- **Friend** (uid, fuid, settime)
uid references uid in User. fuid references uid in User.
- **Neighbor** (uid, nuid, settime)
uid references uid in User. nuid references uid in User.
- **Hoodapp**(uid, hid, confirm)
uid references uid in User. hid references hid in Hood.
- **Thread** (tid, uid, hid, bid title, visibtype, posttime, longitude, latitude)
uid references uid in User. hid references hid in Hood. bid references bid in Block.
- **Message** (tid, no, uid, hid, bid, posttime, text)
tid references tid in Thread. uid references uid in User. hid references hid in Hood. bid references bid in Block.
- **Setting** (uid, emailntf, fntf, nntf, bntf, hntf)
uid references uid in User.
- **Msgbox** (uid, tid, no, mtype, eid, etext, settime)
uid references uid in User. tid references tid in Thread.

IV General Assumptions

Since the project is only a prototype, some assumptions are made:

- Neighborhoods and blocks are predefined in the database. Users are not allowed to create new neighborhoods and blocks.
- Each email can only register one account. And users use emails to login.
- Pictures and hyperlinks are not considered in the prototype. This may relate to the design of user profile and message board.

V Back-end Design

V.1 User Account and Joining

Users should use emails to log into the application. Name and email are not allowed to be modified after sign up. Every time user logs in, lastlogin attribute in User will update to the current time.

Users can join blocks and neighborhoods later after signing up. Also they can modify their profile at anytime. Joining neighborhoods needs 3 people in the neighborhood to confirm. This information is stored in confirm attribute in Hoodapp.

When a user applies to join a neighborhood, msgbox will add notifications to the users in that neighborhood. This record will not be deleted until the user either accepts or rejects the request.

Setting table allows users to choose which types of messages they will be notified when receiving. However, setting does not influence messages they can access when searching messages (only notifications when they login).

V.2 Friend and Neighbor

Friend is a duplicate relation. Users need to send request to others to become friends. Therefore both pair of uid and fuid and the inverse is needed in the database to indicate the successful build up of friendships.

And when a friend application was made, msgbox will update similarly to that of a neighborhood join application.

Neighbor is a single direction relation. Users can search and add neighbors.

V.3 Block and Neighborhood

Blocks and neighborhoods are pre-defined. Each block and neighborhood has a unique id. Each neighborhood can contain several blocks.

Blocks are treated as rectangles. The position and range of a block is specified by 4 parameters: longitude of NE point, latitude of NE point, longitude of SW point and latitude of SW point. The position shown on map will be the center of each block.

Users can join a block freely without permission. However, in order to join a neighborhood, they need permission by 3 other users' who are currently members of that neighborhood.

V.4 Message System

People can send and browse messages in an online forum like system. Users can create topics (threads in the database) with a title and an initial message. The creator can choose the visibility type for his topic, such as visible by neighborhood, by friends or by block. Position information (latitude and longitude) of the topic is optional.

Messages in a certain topic will be numbered according to the post time. In the database, the initial message that the creator of the topic posts will be number 0. And the following messages posted will be numbered in sequence.

VI Sample Queries

In this part some sample queries are listed to build the database and to implement required functions. To verify the queries, some sample data have been added to the database. The queries along with their results are shown blow.

VI.1 Joining

(1) Sign up a new account with name 'G'.

```
1 insert into user(uid, name, email, jointime) values
2 (7, 'G', 'g@gmail.com', '2019-12-01 10:45:00')
```

uid	jointime	name	email	address	bid	hid	profile	noption	lastlogin
1	2018-02-04 10:45:00	A	A@gmail.com	101, Metrotech 1	1	1	I am A	n	2018-02-04 10:45:00
2	2018-02-04 10:45:00	B	B@gmail.com	102, Metrotech 1	1	1	I am B	n	2018-02-04 10:45:00
3	2018-02-04 10:45:00	C	C@gmail.com	101, Metrotech 2	2	1	I am C	n	2018-02-04 10:45:00
4	2018-02-04 10:45:00	D	D@gmail.com	102, Metrotech 2	2	1	I am D	n	2018-02-04 10:45:00
5	2018-02-04 10:45:00	E	E@gmail.com	101, Duffield 1	3	2	I am E	n	2018-02-04 10:45:00
6	2018-02-04 10:45:00	F	F@gmail.com	101, Duffield 2	4	2	I am F	n	2018-02-04 10:45:00
7	2019-12-01 10:45:00	G	g@gmail.com	NULL	NULL	NULL	NULL	NULL	NULL

(2) Join a block whose name is 'Duffield 1'.

```
1 update user
2 set bid=(select bid
3 from block where bname='Duffield 1')
4 where uid=7;
```

(3) Create or edit profile for 'G'.

```
1 alter table user
2 modify profile varchar(45) not null;
```

```

3 update user
4 set profile='My name is G'
5 where uid=7;

```

uid	jointime	name	email	address	bid	hid	profile	nooption	lastlogin
1	2018-02-04 10:45:00	A	A@gmail.com	101, Metrotech 1	1	1	I am A	n	2018-02-04 10:45:00
2	2018-02-04 10:45:00	B	B@gmail.com	102, Metrotech 1	1	1	I am B	n	2018-02-04 10:45:00
3	2018-02-04 10:45:00	C	C@gmail.com	101, Metrotech 2	2	1	I am C	n	2018-02-04 10:45:00
4	2018-02-04 10:45:00	D	D@gmail.com	102, Metrotech 2	2	1	I am D	n	2018-02-04 10:45:00
5	2018-02-04 10:45:00	E	E@gmail.com	101, Duffield 1	3	2	I am E	n	2018-02-04 10:45:00
6	2018-02-04 10:45:00	F	F@gmail.com	101, Duffield 2	4	2	I am F	n	2018-02-04 10:45:00
7	2019-12-03 07:59:07	G	g@gmail.com	NULL	3	NULL	NULL	NULL	NULL

VI.2 Content Posting

(1) Post a new thread which is visible to people in the same block. This will result in message notifications to other users.

```

1 insert into thread(tid, uid, title, visibtype, posttime) values
2 (2, 1, 'Hi', 'b', '2019-12-01 10:50:00');
3 insert into message(tid, no, uid, posttime, text) values
4 (2, 0, 1, '2019-12-01 10:50:00', 'Nice to meet you all');
5 insert into msgbox(uid, tid, no) values
6 ((select uid from user where bid=1), 1, 0);

```

tid	uid	title	visibtype	longitude	latitude	posttime
1	1	First look at this	n	NULL	NULL	2019-12-01 10:58:00
2	1	Hi	b	NULL	NULL	2019-12-01 10:50:00

tid	no	uid	replyno	text	posttime
1	0	1	NULL	This application is interesting	2019-12-01 10:58:00
2	0	1	NULL	Nice to meet you all	2019-12-01 10:50:00

(2) Reply to a message. This will also lead to message notification.

```

1 insert into message(tid, no, uid, posttime, text) values
2 (2, 1, 1, '2019-12-01 10:51:00', 'Welcome');
3 insert into msgbox(uid, tid, no) values
4 ((select uid from user where bid=1), 1, 1);

```

tid	no	uid	replyno	text	posttime
1	0	1	NULL	This application is interesting	2019-12-01 10:58:00
2	0	1	NULL	Nice to meet you all	2019-12-01 10:50:00
2	1	1	NULL	Welcome	2019-12-01 10:51:00

VI.3 Friendship

(1) Add or accept a friend.

```
1 insert into friend(uid, fuid, settime) values
2 (1, 2, current_timestamp);
```

uid	fuid	settime
1	2	2019-12-03 08:18:16

(2) List all current friends.

```
1 select name from user
2 where uid in (select fuid from friend
3 where uid=1)
```

name
B

(3) List all current neighbors.

```
1 select name from user
2 where uid in (select nuid from neighbor
3 where uid=1)
```

name
B
C

VI.4 Browse and Search Messages

(1) List all threads in a user's block feed that have new messages since the last time the user accessed the system.

```
1 select distinct tid from message
2 where posttime > (select lastlogin from user where uid=1)
3 and bid=(select bid from user where uid=1)
```

tid
1
2

(2) List all messages in her friend feed that have unread messages.

```
1 with a(tid, no) as
2 (select tid, no from message
3 where uid in (select fuid
4 from friend
```

```

5 where uid=1)
6 and posttime>(select lastlogin
7 from user
8 where uid=1))
9 select tid , no from msgbox
10 where uid=1
11 and tid=a.tid
12 and no=a.no

```

tid	no
1	0
2	0
2	1

(3) List all messages containing the words “welcome” across all feeds that the user can access.

```

1 with a(bid , hid) as
2 (select bid , hid from user
3 where uid=1)
4 select tid , no from message
5 where contains(text , 'welcome')
6 and (visibtype='h' and hid=a.hid)
7 or (visibtype='b' and bid=a.bid)

```

tid	no
2	1

VII Front-end Design

The website is developed with PHP 7.3.11, MySQL 8.0.17 and Google Map API.

VII.1 Sign up and Login

Users should use emails to log into the application. Name and email are not allowed to be modified after signing up.

Users can join blocks and neighborhoods later after signing up. Also they can modify their profile at anytime.

← → ↻ ⓘ localhost:8000/login.html

Login

email

password

When signing up, the back-end will check whether email is occupied and whether confirming password is same as password. Failures will turn to an exit page.

Sign up

Email:

Name:

Password:

Confirm Password:

Address:

Profile:

You can edit your profile later

This project uses php session to maintain logged in user information such as user id. And in all the following operations and web pages, uid will be firstly retrieved from this session.

```
8 session_start();
9 $uid = $_SESSION["uid"];
```

VII.2 Homepage

User homepage contains many elements.

Home

New Notifications:

F wanted to be your friend

F wanted to join MetroTech

G wanted to be your friend

G wanted to join MetroTech

New Messages

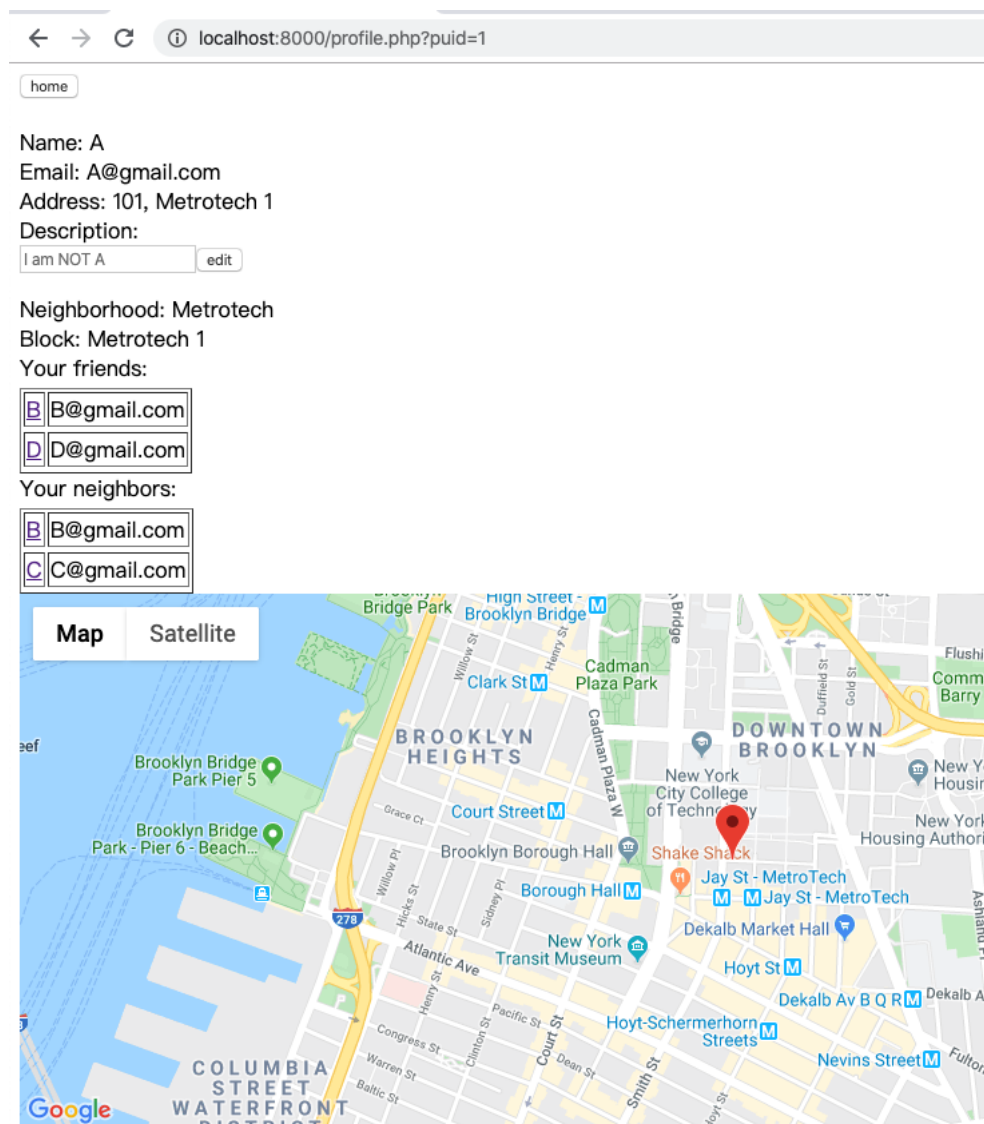
Topic	Message
First user	I am exploring more.

The first row contains 5 buttons which were used to editing profile, modifying settings, searching users, searching blocks and searching neighborhoods. Clicking each of the buttons will turn to corresponding target web page.

Following is new notifications. There are 4 types of notifications: new friend request, joining neighborhood request, new friend added and new neighborhood joined. The request notification will not eliminate until the user either accepts or rejects the request.

Finally there is new message block. Maximum of 10 recent messages will appear here. Users can click the topic of each message to see the whole topic.

VII.3 Profile



puid in the url field is used to pass which profile the user is looking at. Name, email, address, description will be firstly displayed. If the user is looking at his/her own profile, description field can be modified.

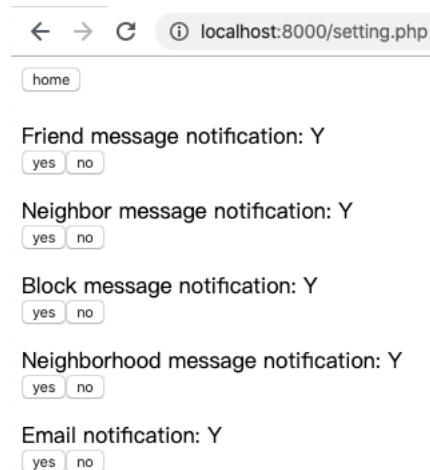
Then there are neighborhood and block field. If the user has friends and neighbors, the information will be shown in the two tables. By clicking the name of

friends or neighbors, user can head to the corresponding profile.

Information of user's friends and neighbors are in his/her profile page. Users can add a neighbor or send friend request on other's profile page.

Location of the block the user lives at is displayed by Google Map API.

VII.4 Setting



← → ↻ ⓘ localhost:8000/setting.php

[home](#)

Friend message notification: Y
[yes](#) [no](#)

Neighbor message notification: Y
[yes](#) [no](#)

Block message notification: Y
[yes](#) [no](#)

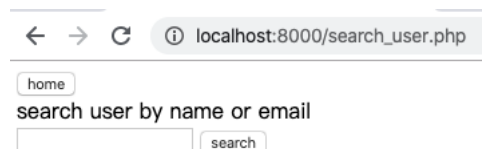
Neighborhood message notification: Y
[yes](#) [no](#)

Email notification: Y
[yes](#) [no](#)

In user setting page, users can modify 5 settings. These 5 settings will influence which messages show in the home page. Since this is only a prototype, email notification has not been implemented.

VII.5 Search

4 searching targets are provided: user, neighborhood, block and message.

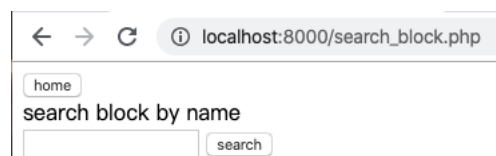


← → ↻ ⓘ localhost:8000/search_user.php

[home](#)

search user by name or email

[search](#)

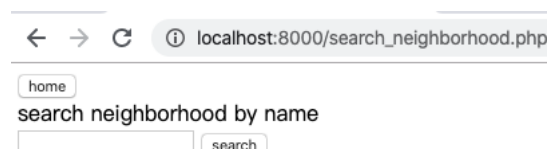


← → ↻ ⓘ localhost:8000/search_block.php

[home](#)

search block by name

[search](#)

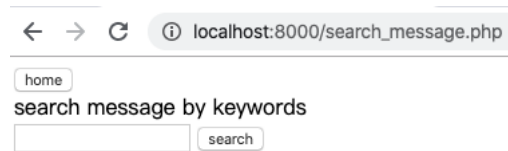


← → ↻ ⓘ localhost:8000/search_neighborhood.php

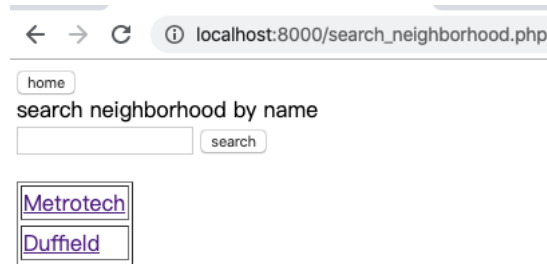
[home](#)

search neighborhood by name

[search](#)

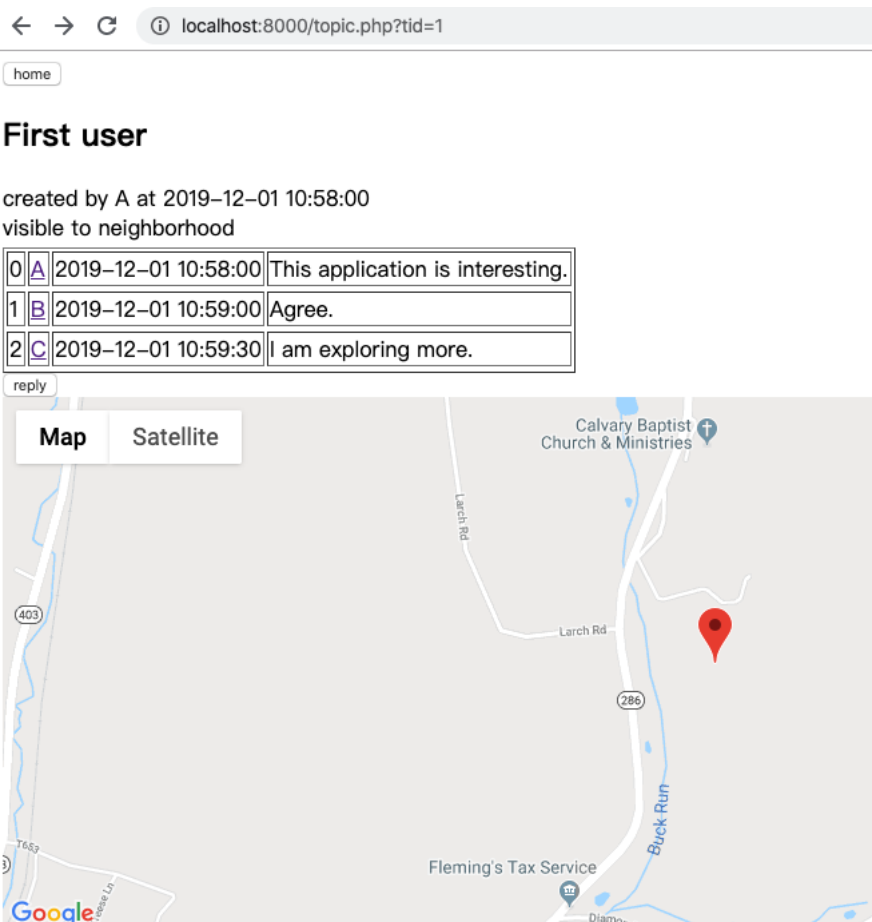
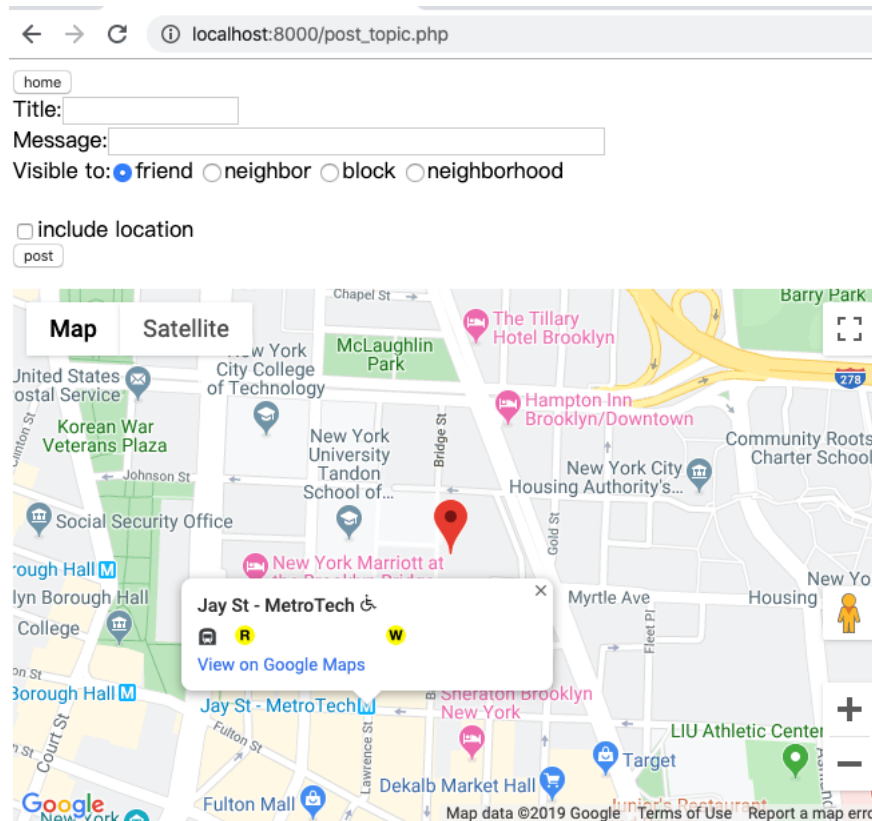


Keyword search methods are implemented by 'like' method in MySQL. Results will be displayed in a table. URLs are added to the names of neighborhoods or users and users can turn to profiles or choose to join a neighborhood/block.



VII.6 Message System

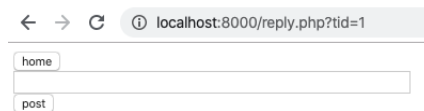
Users can click "post new topic" button on home page to create a topic and post an original message (with number 0). Users can choose the message to be visible to his friends, neighbors, the block or neighborhood he joined. Location is optional. If location block is checked, users can choose location on the map and this will be displayed in topic.



Existing topics can be accessed either by searching messages or by the new messages shown in home page. Topic Information such as creator, post time, visible type and title are displayed at the top. Then there is a table containing all the messages in this topic and the user who posted each. User names contain links to their profiles.

If the topic was created with location information, it will be shown on the map below.

At the bottom, user can click on reply button to reply a message to this topic.



VIII Map API

3 functions are provided to use Google Map API. `createMap()` creates a map at any place. `posMap($latitude, $longitude)` allows to create a map with red bullet at given position specified by latitude and longitude. `clickMap()` creates a map that allows user to click on and return latitude and longitude at the red bullet point.

Note that this is a keyless version of source API functions. You can change the source js link to change it to an official version which needs an Google Map API key.

```
53      src='https://cdn.jsdelivr.net/gh/somanchiu/Keyless-Google-Maps-API@v5.0/mapsJavaScriptAPI.js'>
```

IX Security and SQL Injection

PHP security methods are added to keep secure of user information as well as back-end database.

First "addslashes" method is applied to keep safe of POST method.

```
5      $email = addslashes($_POST['email']);
6      $password = addslashes($_POST['password']);
```

And for all queries involving user inputs, "mysql_real_escape_string" method is used to prevent SQL injections.

```
32      $emailSQL = "select * from user where email = '$email'";
33      $emailSQL = mysql_real_escape_string($databaseConnection, $emailSQL);
34      $resultSet = mysqli_query($databaseConnection, $emailSQL);
```

Also, prepared queries are used to prevent injections. Following is an example in register.php:

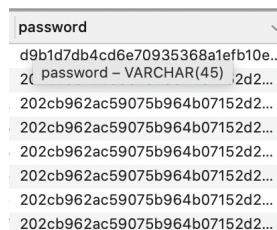
```
21 $query = "insert into users(uid, address, email, password, name ) values(?,?,?,?)";
22
23 $stmt = $db->prepare($query);
24
25 $stmt->bind_param("sssd",$uid,$address,$email,$password,$name);
26
27 $stmt->execute();
```

X File List and Usage

SQL files:

- createTable.sql: Create tables in the database.
- sampleData.sql: Insert sample data to the database.

User passwords are encrypted using MD5 method in MySQL.



PHP function files:

- database.php: Create/close connection to database; Create google map.
 - getConnect(): Connect to database and create a global connection instance \$databaseConnection. Developers should modify hostname, database name, user name and password in this function.
 - closeConnect(): Close connection to database.
 - createMap(): Create and insert a sample map at any position.
 - posMap(\$latitude, \$longitude): Create and insert a map located at given point specified with latitude and longitude.
 - clickMap(\$latitude, \$longitude): Create and insert a map located at given point specified with latitude and longitude. It returns a position of clicking point.

- `fileSystem.php`: Provide uploading file function.

`upload($file, $filepath)`

Web files:

- `login.html`; `login.php`: Login page.
- `register.html`; `register.php`: Sign up page.
- `home.php`: Home page.
- `setting.php`: Setting.
- `profile.php`: Profile.
- `topic.php`: Topic of a thread.
- `reply.php`: Reply message to a topic.
- `post_topic.php`: Create a new topic and post original message.
- `search_block.php`: Search block by keyword in block name.
- `search_user.php`: Search user by keyword in name or email.
- `search_neighborhood.php`: Search neighborhood by keyword in neighborhood name.
- `search_message.php`: Search message by keyword in message or topic title.
- `join_block.php`: Join a block.
- `apply_neighborhood.php`: Apply to join a neighborhood.

To use the project, first open MySQL server and PHP server at the path the source files are put in. Then you can first access `login.html` using a browser. One example of opening a PHP web server is using command: `php -S localhost:8000`.