CS315A : GROUP 05

ONLINE SOCIAL NETWORKING

Arihant Kumar Jain	Ashok Kumar	Sangharsh Aglave
11147	11164	11643
Student Id : 31	Student Id : 28	Student Id : 73
arikj@iitk.ac.in	ashokrm@iitk.ac.in	saglave@iitk.ac.in
IITKanpur	IITKanpur	IITKanpur

April 18, 2014

Abstract

This project aims to develop an online social networking site to connect people around the globe. A social networking site connects people digitally. Our endeavour was to make this connection robust. We took inspiration from Facebook and Quora and created a mashup which has the basic features of both of them and weaved it in a beautiful interface. We have used relational database on ruby on rails for implementation of the social network.

1 Introduction

What is social networking?
Social networking is the grouping of individuals into specific groups and expanding the number of social contacts by making connections through individuals. While social networking has gone on almost as long as societies themselves have existed, the unparalleled potential of the Internet to promote such connections is only now being fully recognized and exploited, through Web-based groups established for that purpose.

What is this project achieving?
This project aims to provide an interactive web-platform for people to connect with each other, share informations, ask questions, chat privately, create pages of public figures. Like-minded people can also create specific groups where they can discuss their views in private or public.

How is this achieved?
We have developed an online social networking site using following technologies:

- **Web framework: Ruby on Rails[4].** We chose ruby on rails over PHP due to following reasons:
  - This framework can accomplish more functionalities with better structured code.
  - Rails is built on Ruby, the dynamic and object-oriented language. For instance, it will provide a template system for handling page sections and layouts, process Ajax updates and a wide set of plugins.
  - Moreover, ruby on rails has inbuilt support of a database which takes care of all database communications efficiently.
- **Database: Relational Database, sqlite3.** In order to optimise query processing and updates in the database, we have ensured that all the relations in the database are in BCNF.
- **Picture uploads: Paperclip[1].** For uploading profile or timeline pictures in the database, file attachment library - Paperclip is used for Active Records in ruby.
- **Front-end development - Twitter Bootstrap[6]** Bootstrap is used to properly layout the front-end templates, button, forms and other interface components.

The code for this project can be found here :
`https://github.com/stranger9811/social-Networking-website/`

2 Approach

The social network designed by us can be modelled in six entity-relationship digrams which are described in detail as follows:

2.1 User - User relation

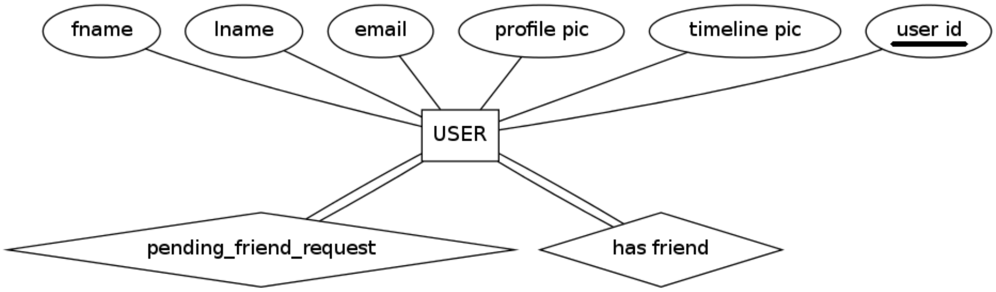

Figure 1: User - User relation

This relation determines how a user is related to other users in the network. There are two ways:

- **pending friend request:** If one user(say,A) sends friend request to other user(say,B), until the user B accepts the request, both users share the **pending friend request** relation as shown in the 1.
- **Friends:** If user B accepts the friend request of user A, then the relationship between them change from pending friends to friends.

2.2 User - Page relation

Just as each user has a profile, a public page can be created for celebrites, organisations, and other entities. Unlike, user-profile, these pages are public, i.e., they are visible to every user. Users can post and read posts from a page once they like them. Users can be related to a page in following ways:

- **Create page:** User can create new pages with attributes like title, description, timeline picture etc.
- **Like page:** User can like any number of pages of their interest and start reading and writing posts on the page.
- **Page posts:** Each page can have multiple posts by different users and users can even comment on the post by other users.

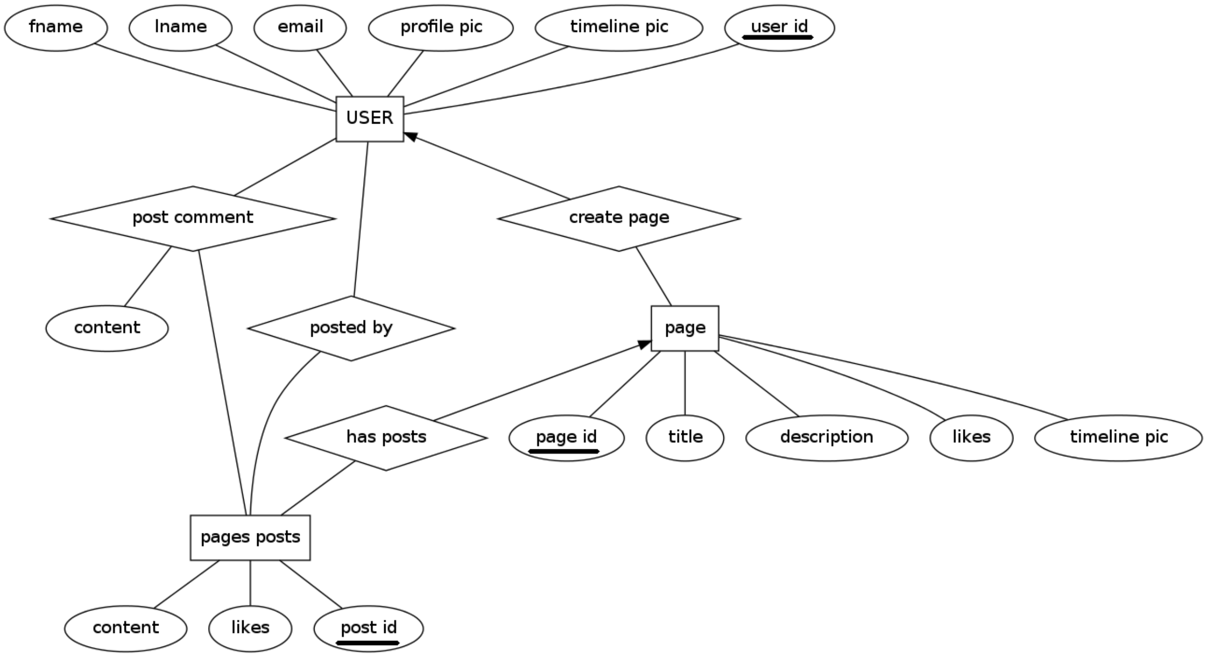

Figure 2: User - Page relation

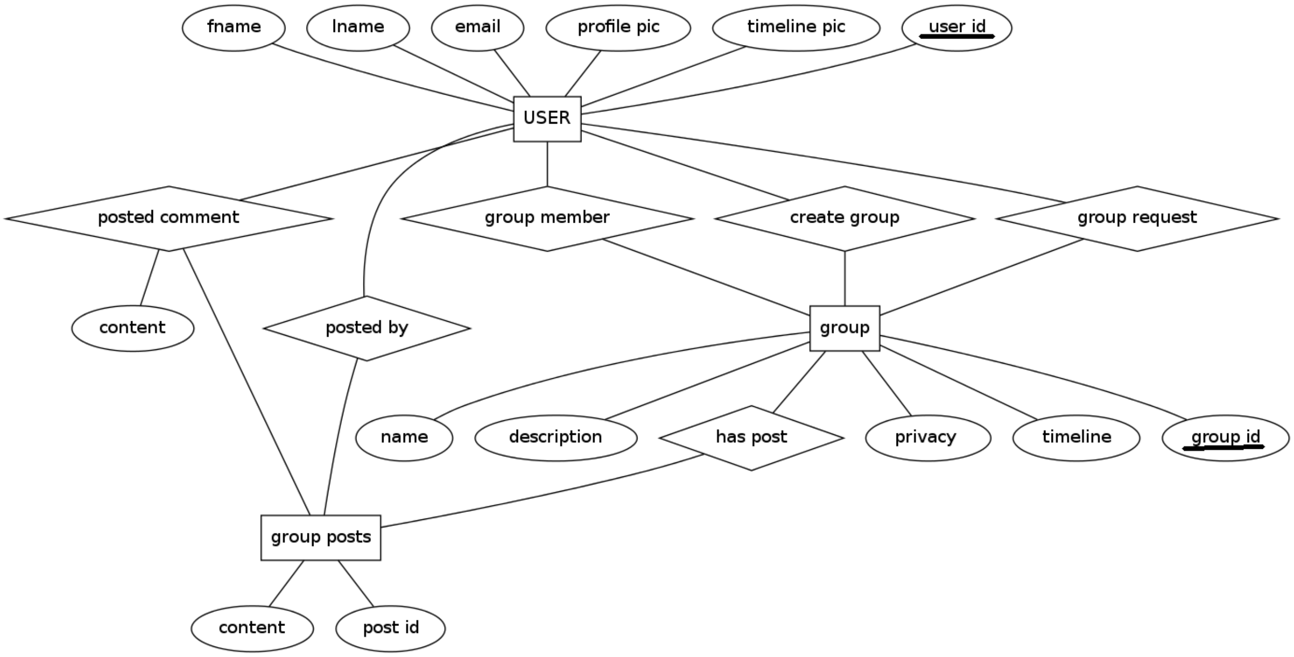

Figure 3: User - Group relation

2.3 User - Group relation

Pages are designed to be the profiles for entities, groups are the place for small group communication and for people to share their common interests and express their opinion. When a new group is created by a user, he/she can decide whether to make it publicly available for anyone to join, or require his/her approval for members to join. The relationship between users and groups are described below:

- **Create group:** A user can create new groups which can be made public or private as described above. Groups will have attributes like title, description, timeline picture, privacy etc.
- **Members:** If a group is public, users can directly become its member. But, if it is private, then a request will be send from the user to the creator of the group. It is upto the creator whether to allow the user to join or not.
- **Group posts:** Only members of groups are allowed to view the content and write new posts in the group.

2.4 User - Question relation

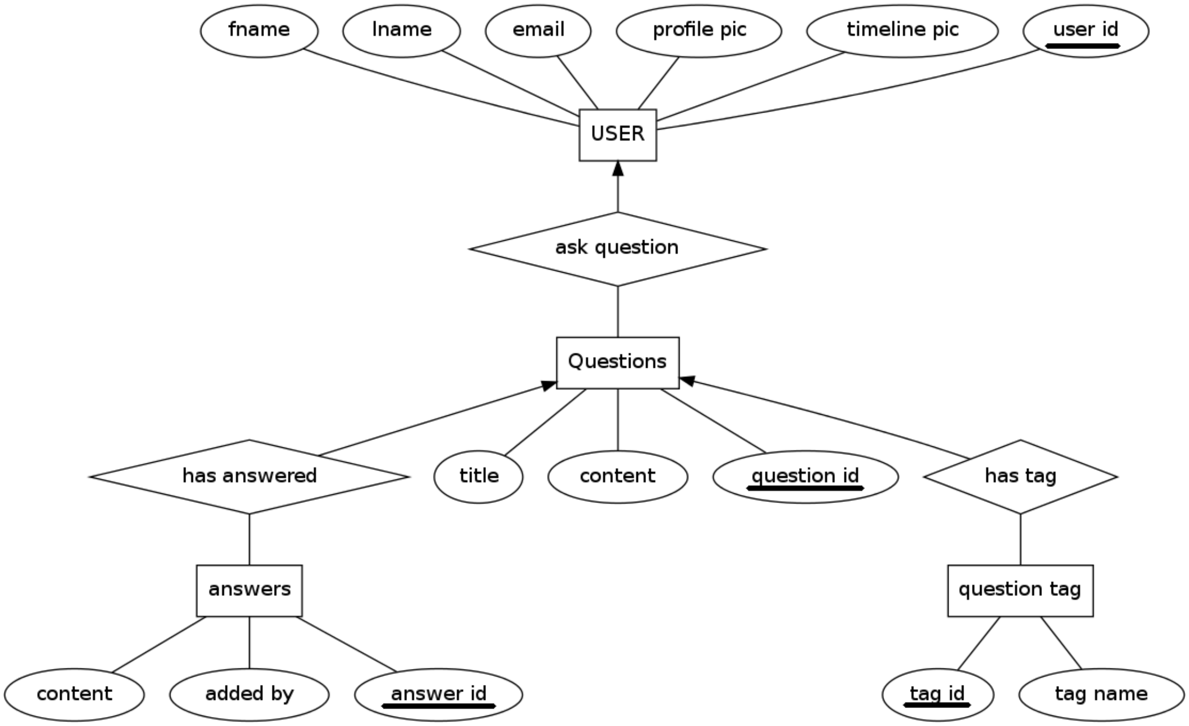

Figure 4: User - Question relation

Apart from pages and groups, a platform is provided to all users to ask question which will be visible to everyone and any user can answer to anyone's query. Users can even search for questions, if already asked by other users and find their answers. User - Question relation can be described as follows:

- **Ask questions:** User can ask any question which would be visible to everybody to answer. A question will have attributes - question title, content.
- **Has answers:** A question can have multiple answers from multiple users. An answer will contain a content and the user who added that answer.

- **Question-tags:** In order to optimize the search query, each question will have certain tags added by user who asked that question.

2.5 User - Wall relation

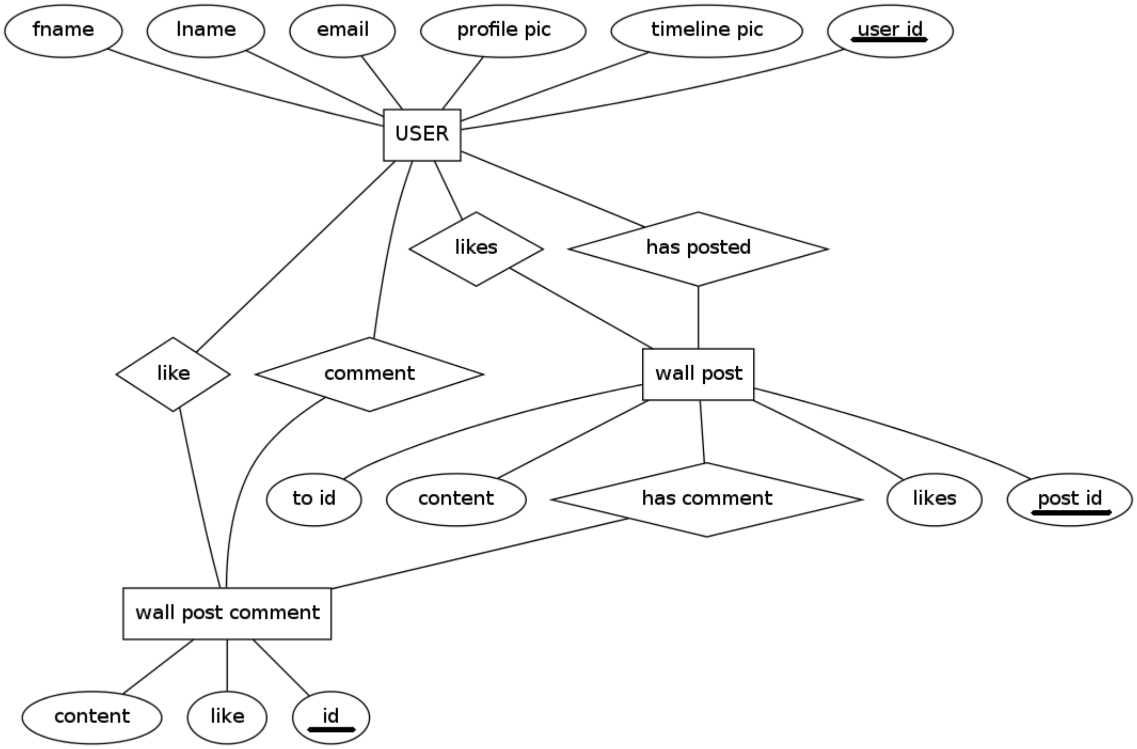

Figure 5: User - Wall relation

Each user has certain area on his profile, which we call, *wall*, where his/her friends can post or comments. User or his/her friends can like each others post or comments on each other walls. User can interact with walls in following way:

- **Wall post:** A user can post to his/her own wall or on the walls of his/her friends. Each post has attributes: content, user on whose wall it is posted, number of likes on the post etc.
- **Wall comment:** Users can even comment on the posts by other users which will have same attributes as posts.
- **Like post/comment:** A user can like any posts and comments.

2.6 User - Message relation

Another interesting feature of the social network is the instant messaging. Users can communicate with each other privately using chat and it will store all the previous conversations. Each message will have some content and only two users can be related through messages.

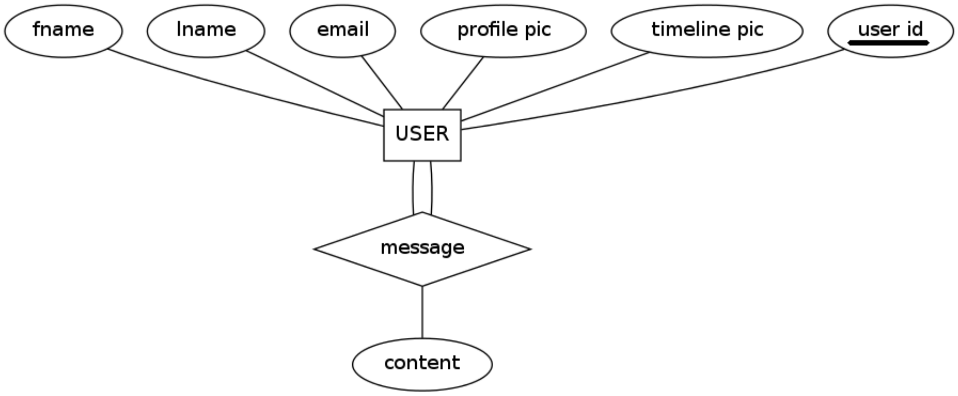

Figure 6: User - Message relation

3 Features

3.1 Security measures

- **Email verification:** At the time of registering new users, we need their email address in order to verify their identity. When users sign up , they automatically receive an email request to verify their address. This email is generated using **sendgrid API**[5]. To verify, they just need to click on the link in the message. This veriifcation link is generated using 15 hexadecimal of random number which is stored in the database for verification.

- **Cookies protection:** Since cookies hold sensitive session information, they need to be verified each time they are used in order to ensure that they have not been tampered. Each time a user login, we store following cookies in the browser:

  - username: Name of the user logged in
  - userid: The id of the user, which acts as primary key.
  - checksum: A 15 hexadecimal long random number

  userid and checksum is also stored in the database on server side and everytime cookies are used, checksum stored in the browser is matched with that stored in the database. If matching fails, user is logged out immediately and need to login again, otherwise, required action is executed. This ensures integrality of cookies.

- **Groups privacy:** Since, groups can be private, a non-member is not allowed to view the content of the group unless his request to join group is accepted by the creator of the group.

- **Password change:** We have also given the feature of changing password in case the password of any user is compromised or user forgets his/her password. The password is sent via email, which can be later changed by user.

3.2 Additional features:

- **Search engine:** We have developed a very user friendly search engine. The search engine keeps on matching the incomplete query filled by user and return all the related responses from the database in the real time using **auto-complete javascript library**[3]. Search query can be categorised as follows:

  - Search people: User can simply type the name of the person and all related results can be seen in the drop-down.
  - Search page: User can search for a page through its title using search query - **page :** $< title >$
  - Search group: User can search for a group through group-name using search query - **group :** $< name >$
  - Search question: User can search for any question asked by some user through its tag or title using search query - **question :** $< title/tags >$.

  In all cases, query is matched from database and top five results are displayed to the user.

- **Advanced search:** Apart from directly searching for people, pages and groups, we have implemented advanced query search which allows user to refine their search using specific queries. Following type of advanced queries are supported by us:
  - friends who live in $< city >$
  - people who live in $< city >$
  - friends who study in $< university >$
  - people who study in $< university >$
  - friends who like $< page >$
  - people who like $< page >$

4 Future work

- The user-interface can be improved.
- We can use ajax more extensively so that we have to refresh pages less often.
- We can implement graph database to improve the model of our database.

References

[1] https //github.com/thoughtbot/paperclip

[2] http //guides.rubyonrails.org/association_basics.html

[3] http //api.jqueryui.com/autocomplete/

[4] http //ruby.railstutorial.org/ruby-on-rails-tutorial-book

[5] http //sendgrid.com/docs/Code_Examples/ruby.html

[6] http //getbootstrap.com/css/