# RamisNetwork

*Installation and technical manual*

Guillem Pagès Puig

Desembre de 2013

# Installation

## Download

Download the latest version from Github:
https://github.com/guillemp/RamisNetwork

## System requirements

- PHP >= 5.4.10
- MySQL >= 5.5.29
- Apache 2

## Installation steps

1. Unzip ***RamisNetwork.zip*** into your httdocs folder
2. Chmod 777 folders *img/**avatars*** and *img/**photos***
3. Create a new database and call it ***RamisNetwork***
4. Import *sql/**RamisNetwork.sql*** into your database
5. Edit ***config.php*** file to put your database credentials
    - DB_USER
    - DB_PASSWORD
    - DB_NAME
    - DB_HOST
6. If you have your RamisNetwork folder not in the root or the folder it's named different than *RamisNetwork*, change this parameters:
    - **config.php** -> ROOT
    - js/**functions.js** -> var root;

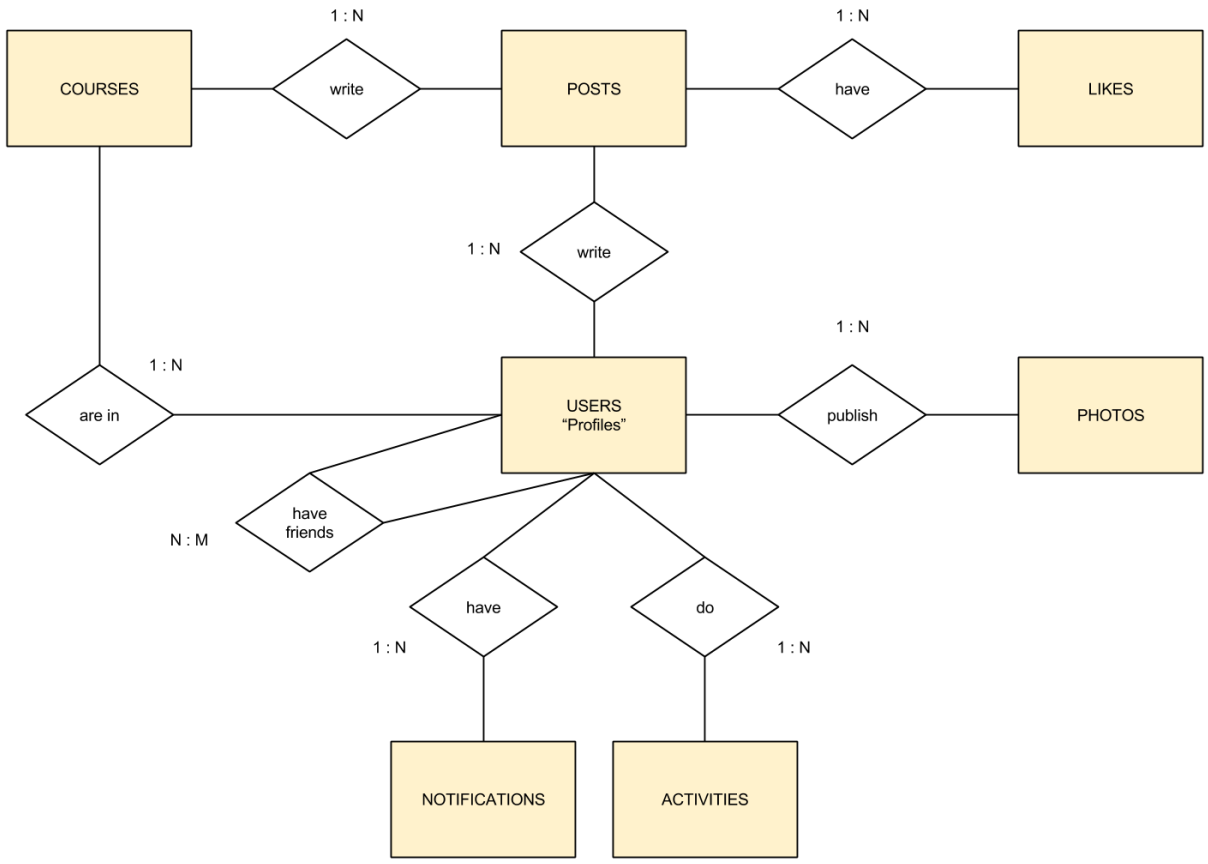
## Testing accounts

All accounts in the network are made to test the correct performance.
You can use any of them using this pattern:

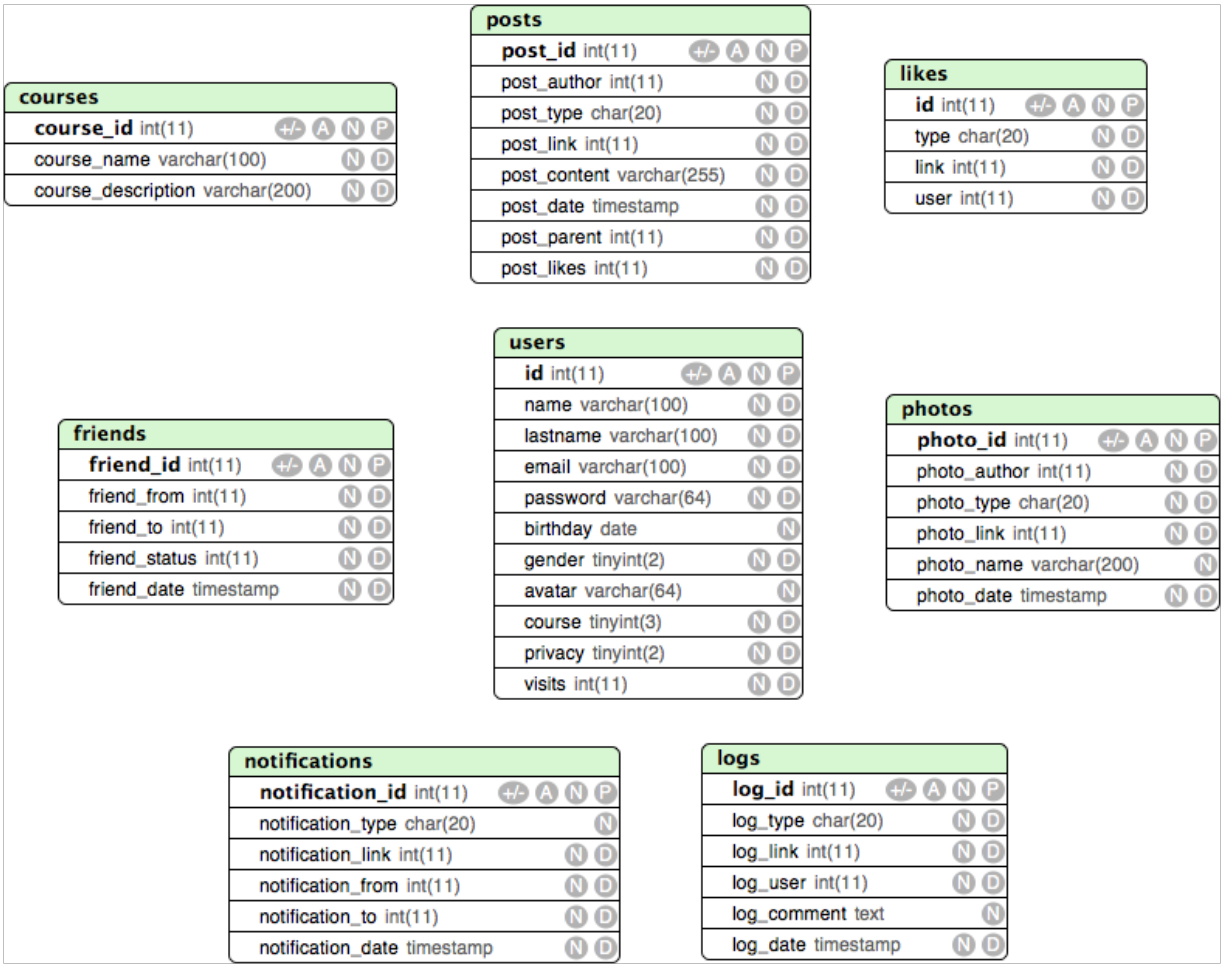
- All the accounts are **name@iesjoanreamis.org**
- The password is four times the first letter of the name. See the example:

    User: **pepe@iesjoanramis.org**
    Password: **pppp**

# E/R diagram


COURSES
1 : N
write
POSTS
1 : N
have
LIKES
1 : N
write
1 : N
are in
USERS
“Profiles”
1 : N
publish
PHOTOS
N : M
have
friends
have
1 : N
do
1 : N
NOTIFICATIONS
ACTIVITIES

# Database fields



# File structure

This application uses the pattern MVC (Model View Controller) and this is how is structurated.

**RamisNetwork**: This is the root folder and it contains folders and the controllers:

- Controllers
  - **course.php**: Shows the course profile
  - **courses.php**: Shows the list of courses and their activity
  - **home.php**: Shows the notifications, friend requests and global timeline
  - **index.php**: Shows login and register buttons
  - **login.php**: Login to the system
  - **members.php**: List, search and filter members
  - **post.php**: Show one individual conversation
  - **profile.php**: The user profile
  - **register.php**: Register to the system
  - **settings.php**: For changing user data and upload profile picture

- Folders
  - **css**: Contains css files
  - **img**: Contains all type of pictures
    - **avatars**: Profile pictures from the users
    - **photos**: Uploaded photos from users
  - **js**: Javascript files are in here
  - **lib**: This is the Libraries folder. It contains models and functions
    - Models
      - **Course.php**: The courses class
      - **db.php**: Database connection
      - **Login.php**: Authentication class
      - **Photo.php**: The photos class
      - **Post.php**: global messages class
      - **User.php**: the user class
    - Other
      - **functions.php**: Contains functions
      - **html.php**: It contains three essential functions
        - do_header(): Loads the header view
        - do_footer(): Loads the footer view
        - do_view(name, data): Loads the views
      - **like.php**: Save like into the post
      - **phpthumb**: To do photo thumbnails
  - **sql**: Database backups
  - **views**: The html views

# Create a new page

1. Create a new controller in the root folder with this structure

```
require('config.php');
require(LIB . 'html.php');

$data['comment'] = "Data for the view";
$data[some_text] = "Some text";

do_header($page_title);
do_view("name", $data);
do_footer();
```

2. Create a view and call it **name_view.php** in the views folder

   To use the data you only have to use the array names.
   The function do_view will convert the array in a separate variables.

```
<p><?php echo $comment; ?></p>
<p><?php echo $some_text; ?></p>
```

3. Create a new model or use the existent ones (User, Post, Course or Photo).

4. To control the **user session** you can use a global variable called `$current_user`

   Current user attributes:

```
$current_user->id; // int
$current_user->name; // string
$current_user->avatar; // string
$current_user->authenticated; // boolean
```

   Check if user is logged in:

```
if ($current_user->authenticated) {
      // user logged in
} else {
      // anonymous user
}
```

## Things to improve

1. Add security in the cookie that stores the session.
2. Different levels of privacy
3. Private messaging
4. Recovery password system
5. Upload photos in courses
6. Notifications via email
7. Send email when register to activate the account
8. Backend to manage the network