# Project Report

## An Online Forum and its Database

Prepared for CPSC 471 Database Management Systems

Team Members:

Kimalel Tuitoek,
Eric Vachon,
Daniel Bezdek

## Table of Contents

# 1. Introduction

## 1.1 Motivation

In this note, we present the development summary, design details, and implementation of an online forum website and its database. The forum aims to straightforwardly facilitate user discussion in an organized, moderated manner.

The previous decade saw a paradigm shift in the organization and collaborative structure of the World Wide Web. In addition to merely absorbing the content of webpages, users are involved in the information exchange processes. Users seek to associate with one another in order to form large, organically evolving virtual communities. In order to accommodate this new requirement, many websites invite the visitor to partake in meaningful content creation. The simplest way to accomplish this is to build a commenting feature into the website.

With any commenting interface, one will naturally encounter the question of how to efficiently orchestrated the information exchange between users. In particular, if we admit a large number of commenters in the discussion, the information exchange problem becomes especially severe. That is, if left unmanaged discussion can quickly devolve into unruly, disorganized dialect! The IT community's technical response to this problem was the introduction of forums. Forums force an overarching order on user interaction by introducing a tree-like directory structure. User posts must belong the specific threads, which articulate specific topics. Furthermore administrators continually scan the forum for policy violators. This allows users to quickly identify topics of interest, and initiate communication with like-minded users. Similarly, administrators can quickly grasp the context of the discussion and remove potential policy violators.

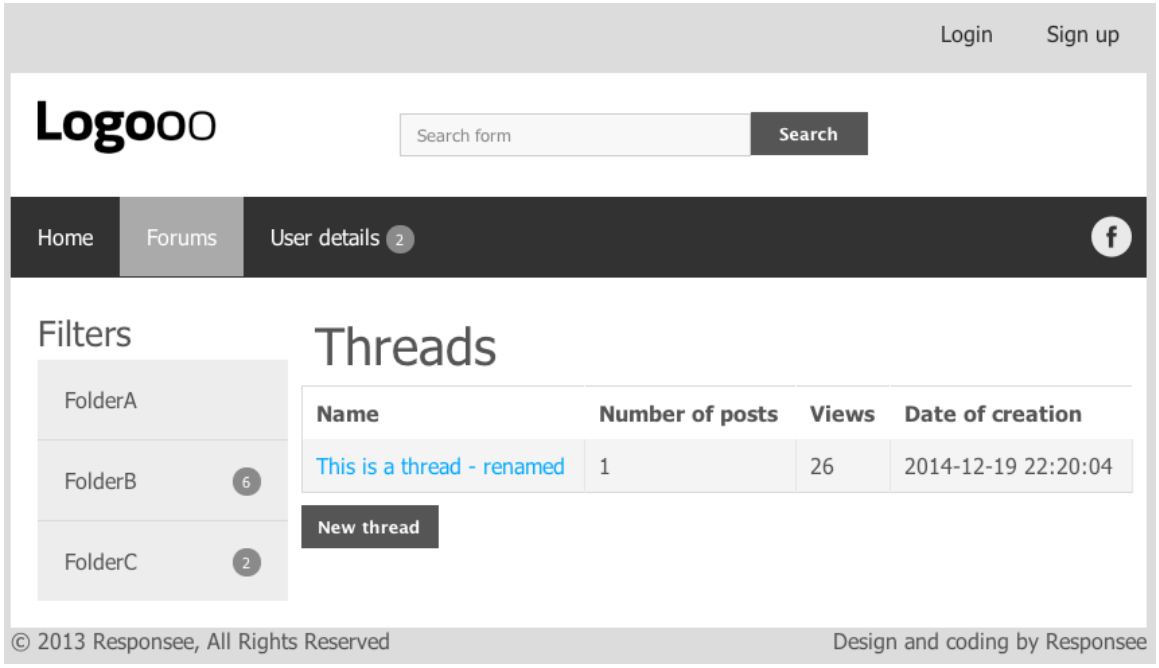
Because the capability to manage forums is vital to any Web 2.0 application, one is naturally interested in finding out how to build one. Since this work inherently entails a database design, we believe it properly addresses the course project requirement.

## 1.2 Overview

Our forum follows the following general organization. We divide the user group of the forum into two distinct groups based on a privilege system. The term **User** shall refer to any customer entity that intends to contribute content to the site. In order to do so, users must create a profile via the front-end environment, which will be subsequently recorded in the database. Once registered, users will have the ability to write posts. By a **Post**, we

mean a block of text generated by a user that is visually encapsulated in field on the forum interface. Generally, the username and the date of the post are also displayed. A number of posts can be organized into a **Thread.** Threads are also user generated, and they correspond to specific topic a user wants to elaborate on. If the topic captures another user's interest, he/she can add post to the thread with the given topic. Similar to other forums, our target product's topics, threads and posts shall assume the form of a tree-structure directory. Users with an extended set of privileges shall be called **Administrators.** The purpose of administrators is to enforce the forum policy in an efficient, decentralized manner. To this end, they shall be equipped with a variety of tools such as the ability to edit and ban posts. It is assumed that administrators are frequent users of the forum, and therefore will be able to respond to the presence of inappropriate content in a timely manner.

## 1.3 Website preliminary




The webpage is designed with usability in mind. In order to accomplish this, we initiated the creation process by studying the typical layout of a well-functioning forum. It is customary to place the login application in upper right corner of the interface, while the search feature is located in the center. Threads occupy the center-right fraction of the screen, to emphasize vitality. The filters occupy the place where directories are usually located in most contemporary graphical user interfaces. The site title and search feature are separated from the lower part of the website with a continuous line of contrasting color to quickly allow the user's eye to categorize the different objects.

A test version of the website can be found at 75.155.72.50:8080

## 1.4 Ergonomic Analysis

Since the interface is eventually targeted at human user, designing the website with human psychology in mind is usually recommended. Since we also faced time pressure in constructing the site design, we wanted to meet the above requirement with the minimum number of visual features possible. We attempted to follow good practice with respect to the Gestalt theory of perception. In particular, we wanted to satisfy: 1) Law of Past Experience-Since the human mind categorizes stimuli according to past experience, replicating a legacy forum outline minimizes user response time. 2) Law of Proximity-Items grouped closely together a perceived to constitute a whole; therefore we minimized average distance between similar objects such as thread lines and folder titles.

## 1.5 Requirements

Here we would like to state the explicit technical requirements that the forum application must satisfy.

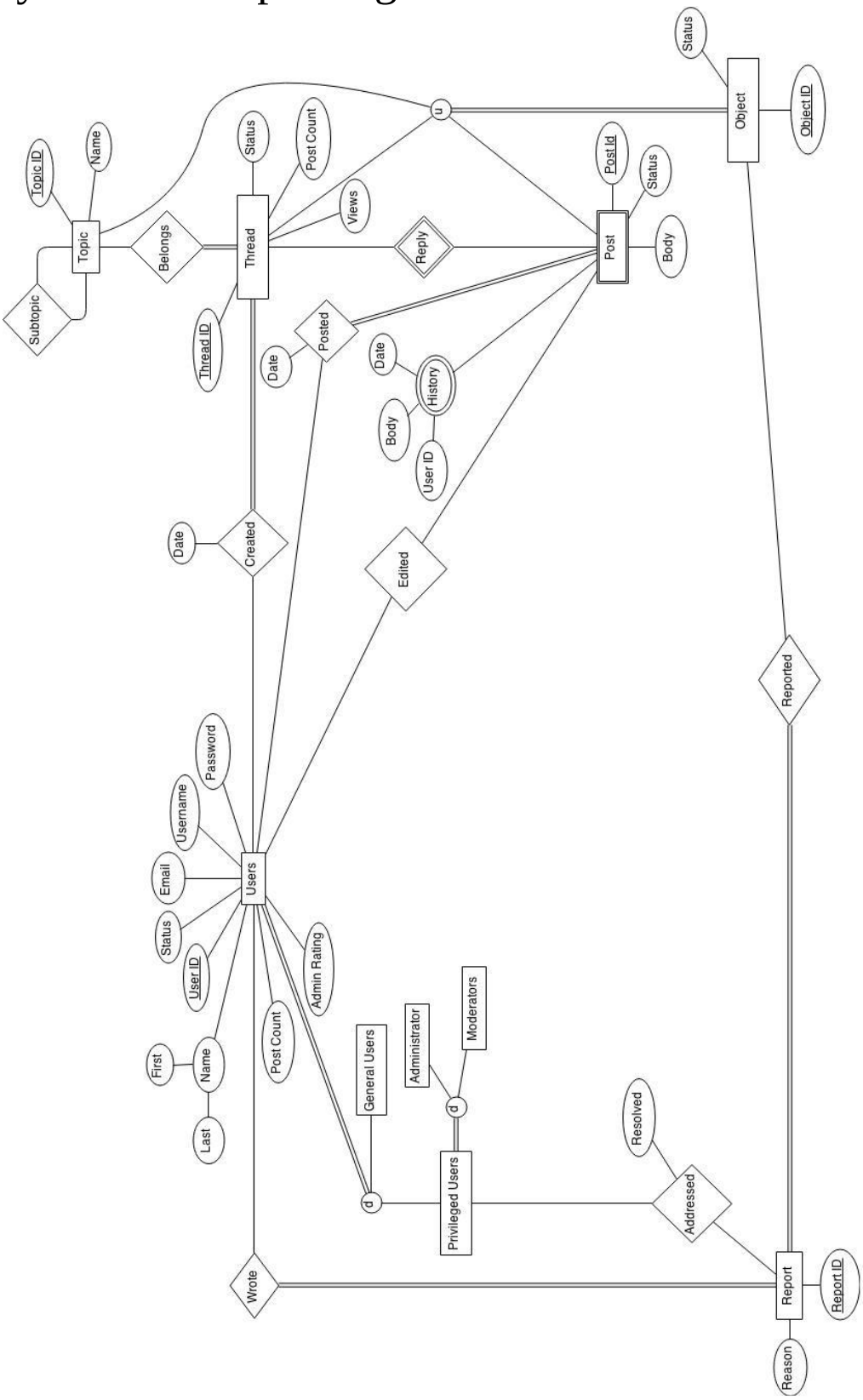
User Requirements

1. All users must be able to create accounts
2. All accounts must be accessible via a log-in feature
3. If they wish, users must be able to edit profile pages
4. Users must be able to create a thread
5. A search functionality has to assist the user to finding threads with appropriate keywords
6. Some of the threads can be labeled private or public in order to limit content visibility
7. A user must be able to add new posts to existing threads
8. An edit feature must allow users to edit their own threads
9. The users should be able to rename their own threads

Administrator Requirements

10. Administrators must have all user functionalities available to them
11. Administrators should be able to edit any post they wish
12. A thread management facility must be available to the administrators. Namely, they must be able to lock, rename, and delete threads.
13. Administrators have to right to interfere with site management hierarchy: they can promote common users to administrator status and demote administrators to users.

# 2. Entity Relationship Design



# 3. General Structure of the Application Program

## 3.1 Activity Diagram for the user

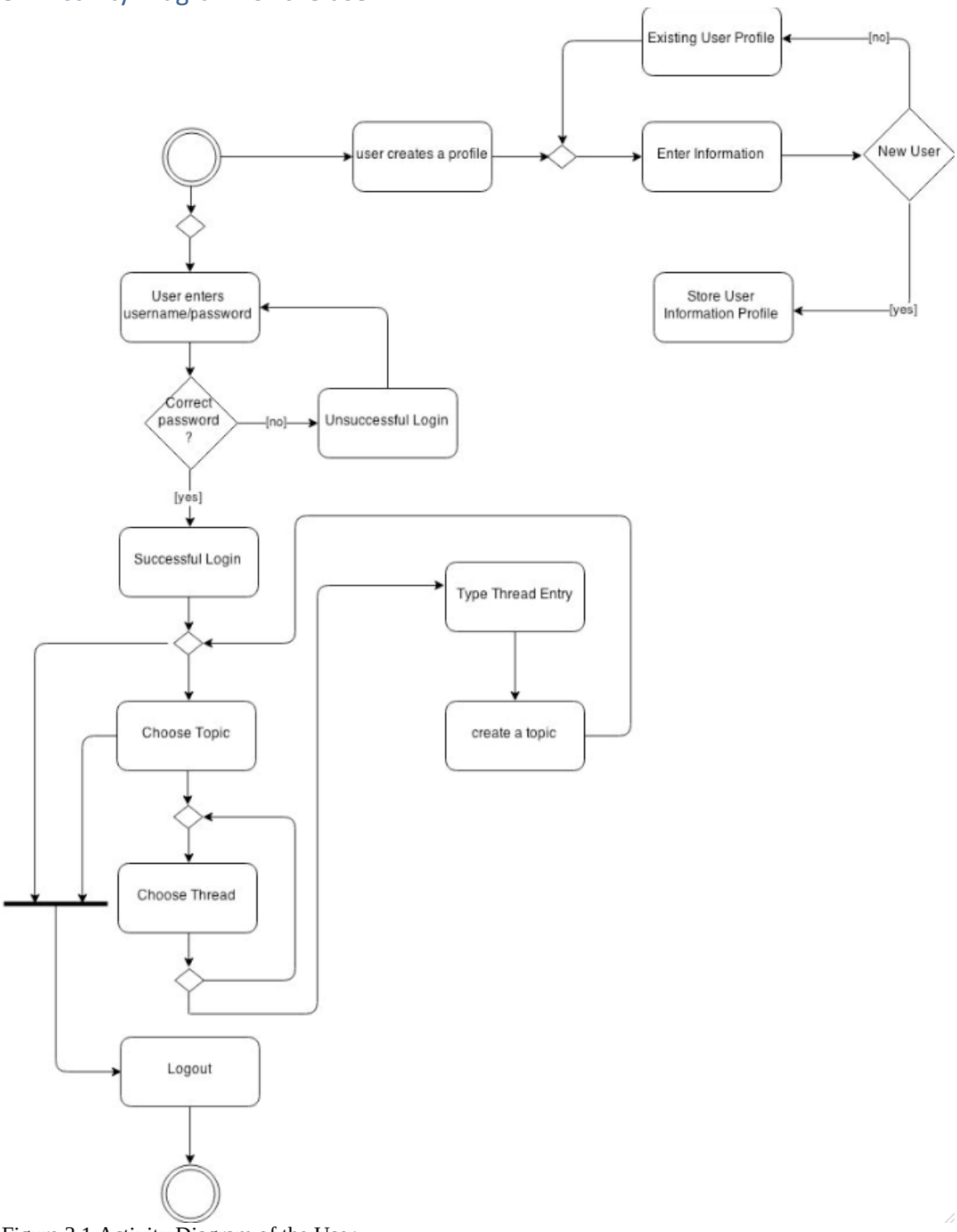



Figure 3.1 Activity Diagram of the User

## 3.2 Activity Diagram for the Administrator

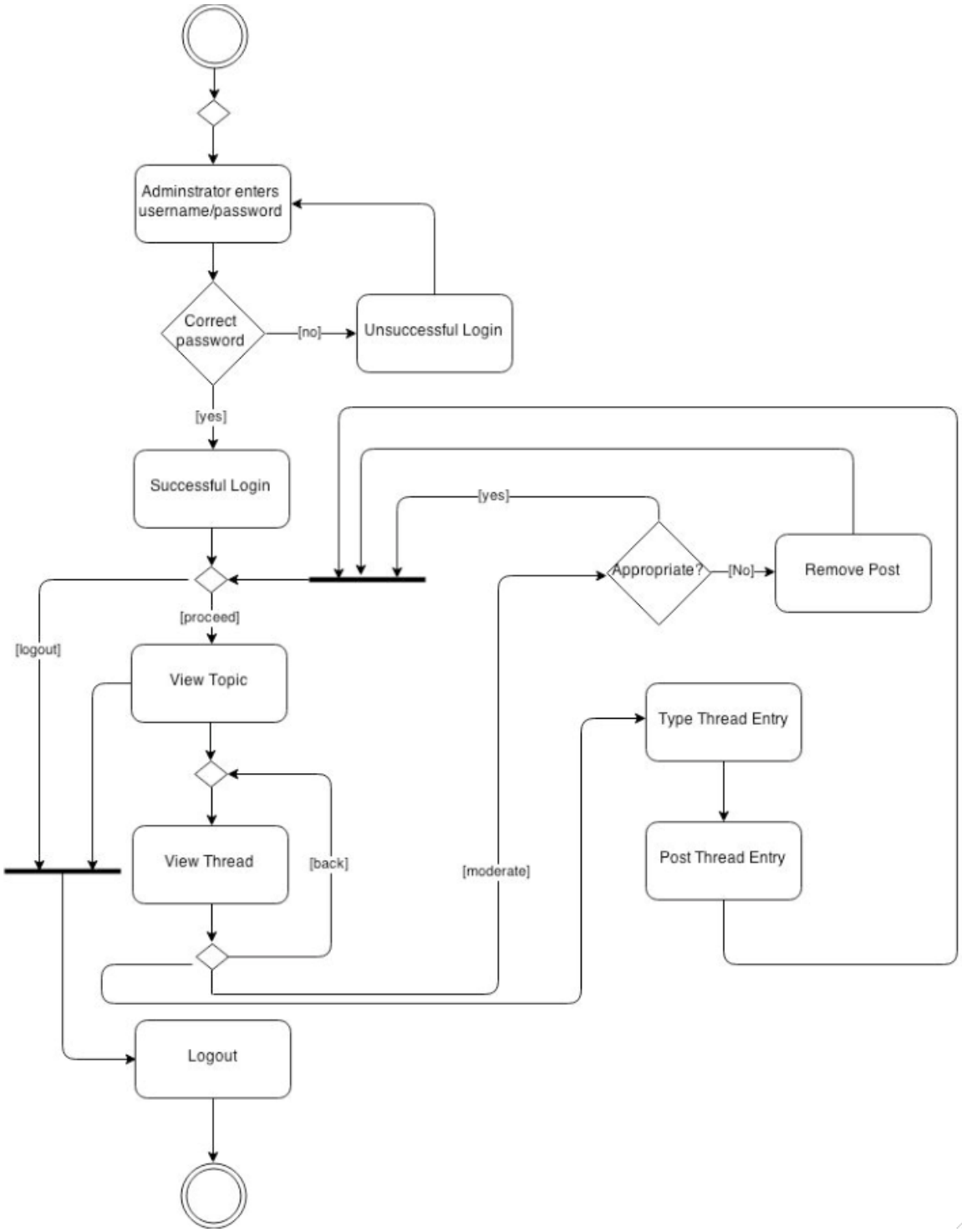


Figure 3.2 Activity Diagram of the Administrator

## 3.3 HIPO Diagram

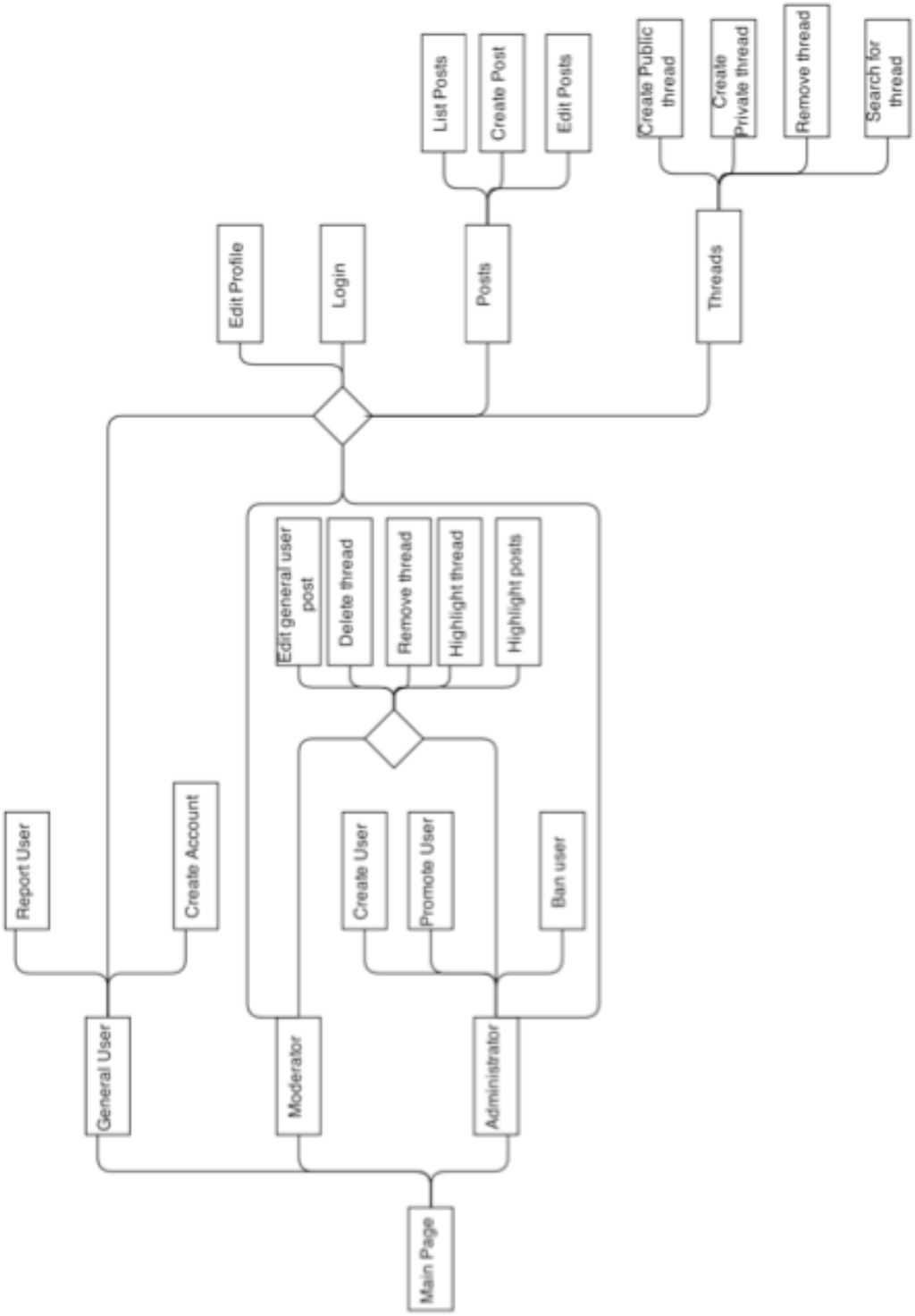



Figure 3.3 HIPO Diagram

## 3.4 HIPO Functions

**General User Module**

| **Function:** | **Create account** |
|---|---|
| **Inputs:** | @Fname, @Lname, @Email, @Username,, @Password |
| **Outputs:** | None |
| **Pseudocode:** | Connect to the database<br>Query = INSERT INTO User(@Fname, @Lname, @Email, @Username, @Password);<br>Parse Query<br>Execute Query<br>Close connection to the database |

| **Function:** | **General User Login** |
|---|---|
| **Inputs:** | @Username, @Password |
| **Outputs:** | UserId,Fname, Lname, Email, Username, Password |
| **Pseudocode:** | Connect to the database<br>Query = SELECT *<br>FROM User as u<br>WHERE (u.Username = @Username) AND (u.Password = @Password);<br>Parse Query<br>Execute Query<br>Close connection to the database |

| **Function:** | **General User Edit Profile** |
|---|---|
| **Inputs:** | @Fname, @Lname, @Email, @Username, @Password |
| **Outputs:** | None |
| **Pseudocode:** | Connect to the database<br>Query = UPDATE User<br>SET “Fname, Lname, Email, Username, Password”<br>WHERE (Fname = @Fname,) AND (Lname = @Lname) AND (Email = @Email)<br>(Username = @Username) AND (Password = @Password); |

| | |
|---|---|
| | Parse Query<br>Execute Query<br>Close connection to the database |

| **Function:** | **Report User** |
|---|---|
| **Inputs:** | @User_Id1, @User_Id2, @Reason, @Thread_Id, @Post_id |
| **Outputs:** | None |
| **Pseudocode:** | Connect to the database<br>Query = INSERT INTO Report(@User_Id1, @Use_Id2, @Reason, @Thread_Id, @Post_id);<br>Parse Query<br>Execute Query<br>Close connection to the database |

| **Function:** | **General User Search for thread** |
|---|---|
| **Inputs:** | @Topic_name |
| **Outputs:** | @Thread_id |
| **Pseudocode:** | Connect to the database<br>Query = SELECT * FROM Thread as t, Topic as tp<br>WHERE (Tp.name LIKE @Topic_name . '%') OR<br>(Tp.name LIKE '%'.@Topic_name) OR<br>(Tp.name Like '%'.@Topic_name.'%');<br>Parse Query<br>Execute Query<br>Close connection to the database |

**Moderator/Administrator module**

| **Function:** | **Moderator Edit general user post** |
|---|---|
| **Inputs:** | @User_Id, @Post_Id, @Body |
| **Outputs:** | None |
| **Pseudocode:** | Connect to the database<br>Query = UPDATE Post<br>SET "Body"<br>WHERE (User_Id = @User_Id,) AND (Post_Id = @Post_Id) AND (Body = @Body); |

| | |
|---|---|
| | Parse Query<br>Execute Query<br>Close connection to the database |

| **Function:** | **Delete thread** |
|---|---|
| **Inputs:** | @Thread_Id, @Name |
| **Outputs:** | None |
| **Pseudocode:** | Connect to the database<br>Query = DELETE FROM Thread<br>WHERE (Thread_Id = @Thread_Id) AND (Name = @ Name);<br>Parse Query<br>Execute Query<br>Close connection to the database |

| **Function:** | **Rename thread** |
|---|---|
| **Inputs:** | @Thread_Id, @New_name |
| **Outputs:** | None |
| **Pseudocode:** | Connect to the database<br>Query = UPDATE Thread<br>SET “Name”<br>WHERE (Thread_Id = @Thread_Id) AND (Name =@New_name);<br>Parse Query<br>Execute Query<br>Close connection to the database |

| **Function:** | **Highlight thread** |
|---|---|
| **Inputs:** | @Thread_Id, |
| **Outputs:** | Thread_Id, Name, Type, Views, Post_count |
| **Pseudocode:** | Connect to the database<br>Query = SELECT *<br>FROM Thread<br>WHERE (Thread_Id = @Thread_Id) AND (Name = @Name);<br>Parse Query<br>Execute Query<br>Close connection to the database |

| **Function:** | **Highlight Post** |
|---|---|
| **Inputs:** | @Post_Id, User_Id |
| **Outputs:** | Post_Id, Body, Date,User_Id |
| **Pseudocode:** | Connect to the database<br>Query = SELECT *<br>FROM Post<br>WHERE (Post_Id = @Post_Id)<br>AND (User_Id = @User_Id);<br>Parse Query<br>Execute Query<br>Close connection to the database |

**Administrator module**

| **Function:** | **Demote user** |
|---|---|
| **Inputs:** | @User_Id, @Username, @Type |
| **Outputs:** | None |
| **Pseudocode:** | Connect to the database<br>Query = UPDATE User<br>SET "Type"<br>WHERE (User_Id = @User_Id)<br>AND (Username = @Username) AND<br>(Type = @Type;<br>Parse Query<br>Execute Query<br>Close connection to the database |

| **Function:** | **Promote user** |
|---|---|
| **Inputs:** | @User_Id, @Username, @Type |
| **Outputs:** | None |
| **Pseudocode:** | Connect to the database<br>Query = UPDATE User<br>SET "Type"<br>WHERE (User_Id = @User_Id)<br>AND (Username = @Username) AND<br>(Type = @Type;<br>Parse Query<br>Execute Query<br>Close connection to the database |

| **Function:** | **Ban User** |
|---|---|
| **Inputs:** | @User_Id, @Username, @Status |
| **Outputs:** | None |

| | |
|---|---|
| **Pseudocode:** | Connect to the database<br>Query = UPDATE User<br>SET “Status”<br>WHERE (User_Id = @User_Id) AND (Username = @Username) AND (Status = @Status);<br>Parse Query<br>Execute Query<br>Close connection to the database |

**Forums module**

| **Function:** | **List posts** |
|---|---|
| **Inputs:** | @Thread_id |
| **Outputs:** | Post_Id, Body, Date, User_Id |
| **Pseudocode:** | Connect to the database<br>Query = SELECT *<br>FROM Post<br>WHERE (Thread_Id = @Thread_Id);<br>Parse Query<br>Execute Query<br>Close connection to the database |

| **Function:** | **Rename thread** |
|---|---|
| **Inputs:** | @Thread_Id, @New_name |
| **Outputs:** | None |
| **Pseudocode:** | Connect to the database<br>Query = UPDATE Thread<br>SET “Name”<br>WHERE (Thread_Id = @Thread_Id) AND (Name =@New_name);<br>Execute Query<br>Close connection to the database |

| **Function:** | **Create Public thread** |
|---|---|
| **Inputs:** | @Name, @Type |
| **Outputs:** | None |
| **Pseudocode:** | Connect to the database<br>Query = INSERT INTO Thread(@Name, @Type = “Public”);<br>Parse Query<br>Execute Query |

| | |
|---|---|
| | Close connection to the database |

| **Function:** | **Create Private thread** |
|---|---|
| **Inputs:** | @Name, @Type |
| **Outputs:** | None |
| **Pseudocode:** | Connect to the database<br>Query = INSERT INTO Thread(@Name, @Type = “Private”);<br>Parse Query<br>Execute Query<br>Close connection to the database |

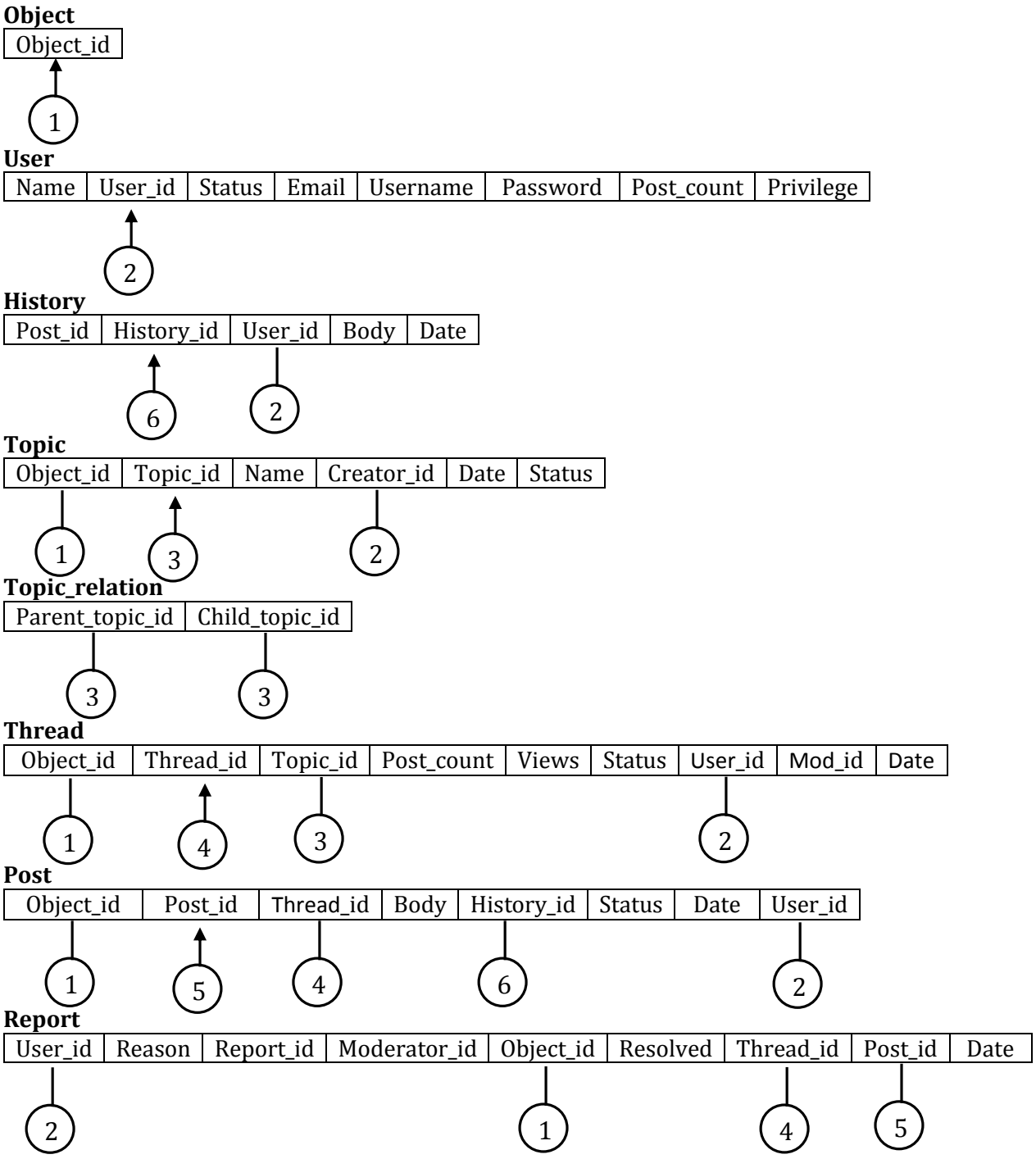
| **Function:** | **Create Post** |
|---|---|
| **Inputs:** | @User_Id, @Body |
| **Outputs:** | None |
| **Pseudocode:** | Connect to the database<br>Query = INSERT INTO Post(@User_Id, @Body);<br>Parse Query<br>Execute Query<br>Close connection to the database |

| **Function:** | **Edit Post** |
|---|---|
| **Inputs:** | @User_Id, Post_Id, @Body |
| **Outputs:** | None |
| **Pseudocode:** | Connect to the database<br>Query = UPDATE Post<br>SET “Body”<br>WHERE (User_Id = @User_Id,)<br>AND (Post_Id = @Post_Id) AND (Body = @Body);<br>Parse Query<br>Execute Query<br>Close connection to the database |

# 4. (Relational) Logical Database Design

## 4.1 Relational Data Model

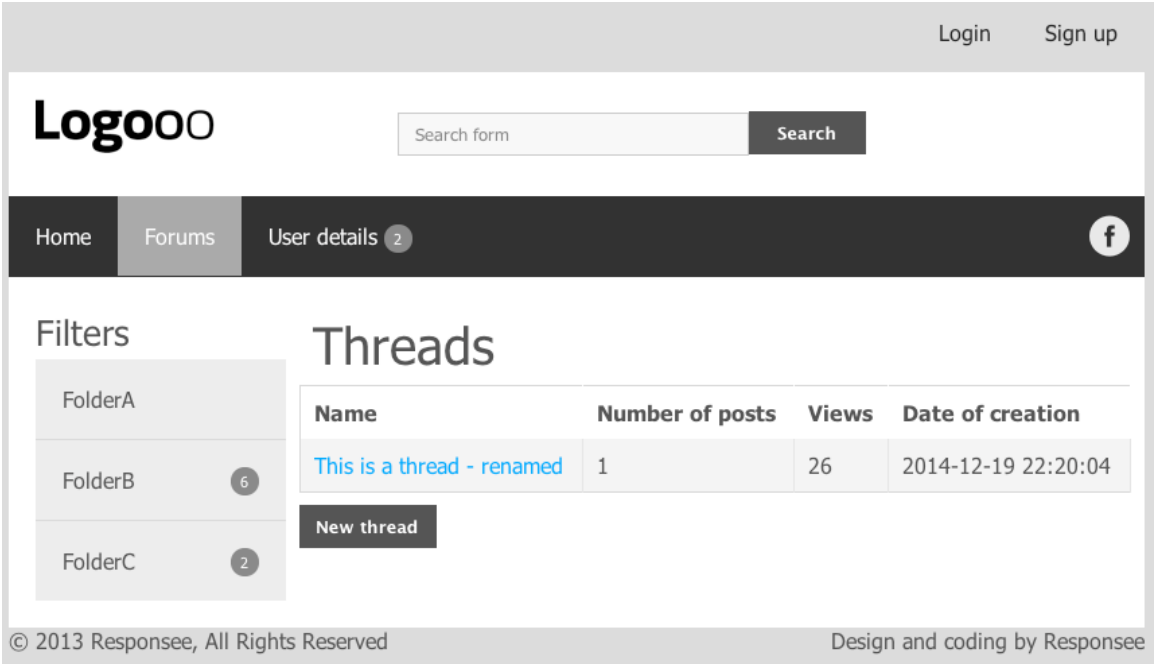
The data structures found in the database are as follows, with the numbers being used to convey the nature of the foreign relations without cluttering the diagram.



# 5. User Manual for Front End Application

## 5.1 The Site

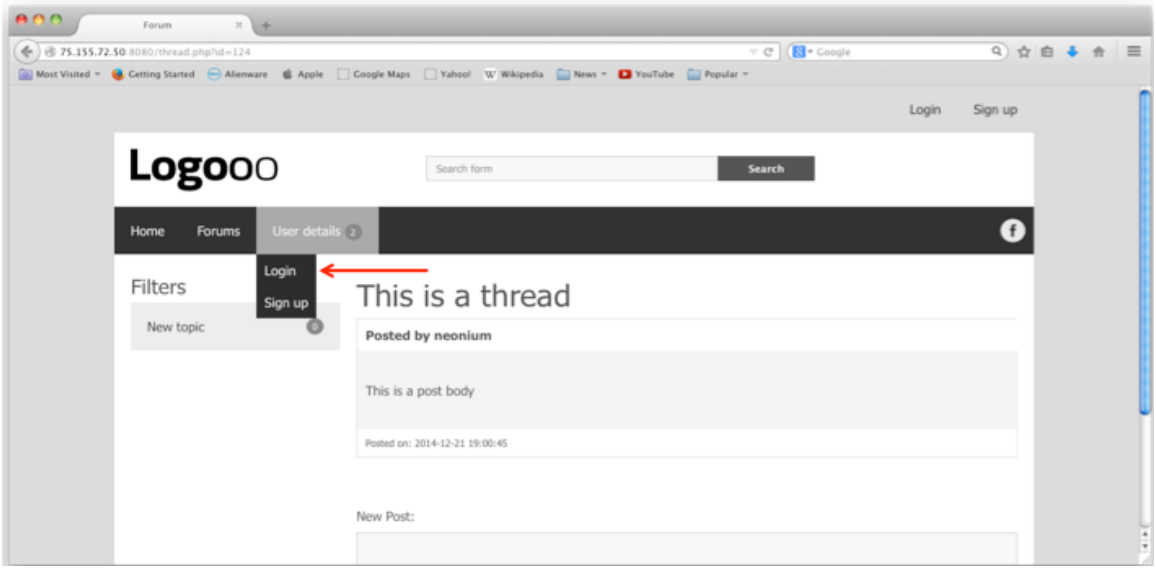
Upon initializing the website, one is presented with the following forum start page




Editing and thread creating functions are available to users and administrators of the site only. Therefore a mandatory login protocol is in effect.

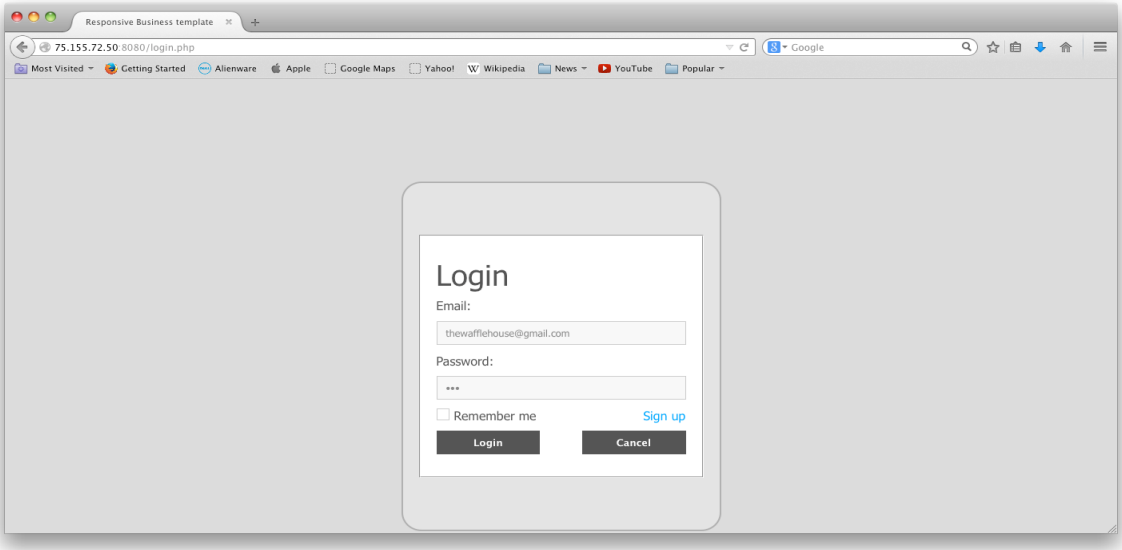
## 5.2 User details

A visitor can choose from two distinct login procedures: either via the standard login in the upper right corner, or through user details as the red arrow indicates in the image below.



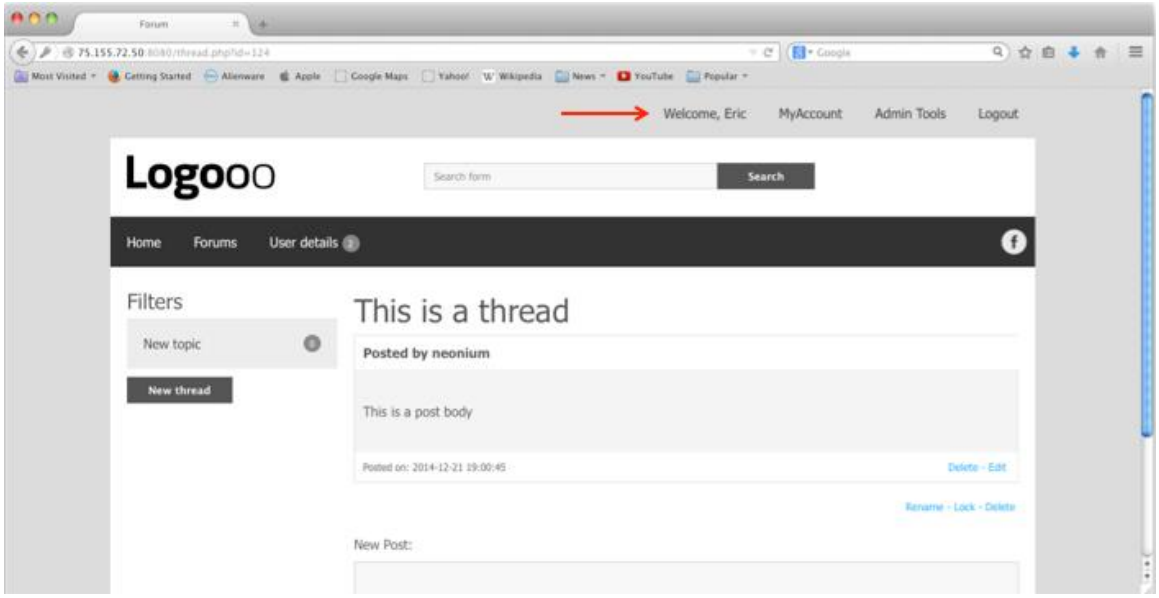

## 5.3 The Login page

The site visitor is navigated to a login page where he/she is presented with the opportunity to login.



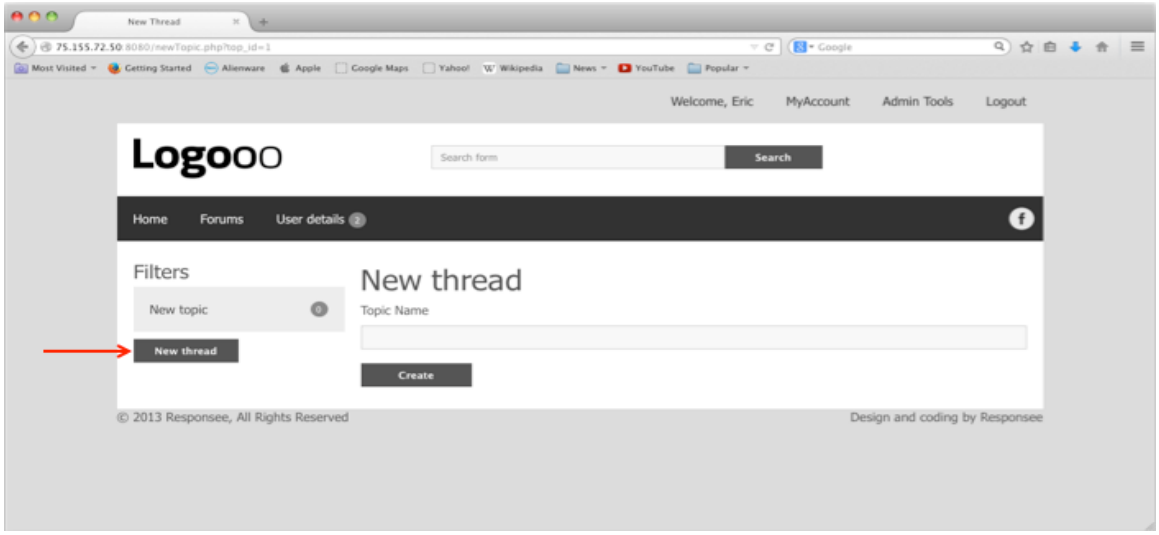

## 5.4 The User Homepage

Once the user's login information is validated, he/she is navigated to a homepage. The upper right corner indicates the current site status. The red arrow in the image below shows a number of user functionalities, such as a tool to edit the user profile.



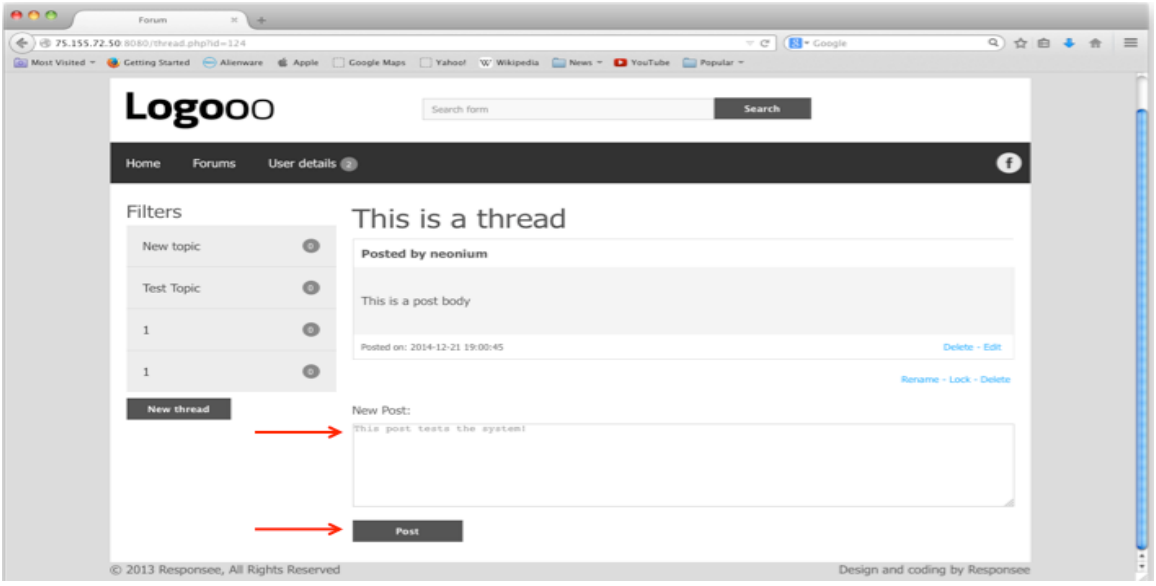

## 5.5 Creating a New Thread

A new thread can be created by clicking on the aptly name button marked by the red arrow. The system will request a topic name from the user.



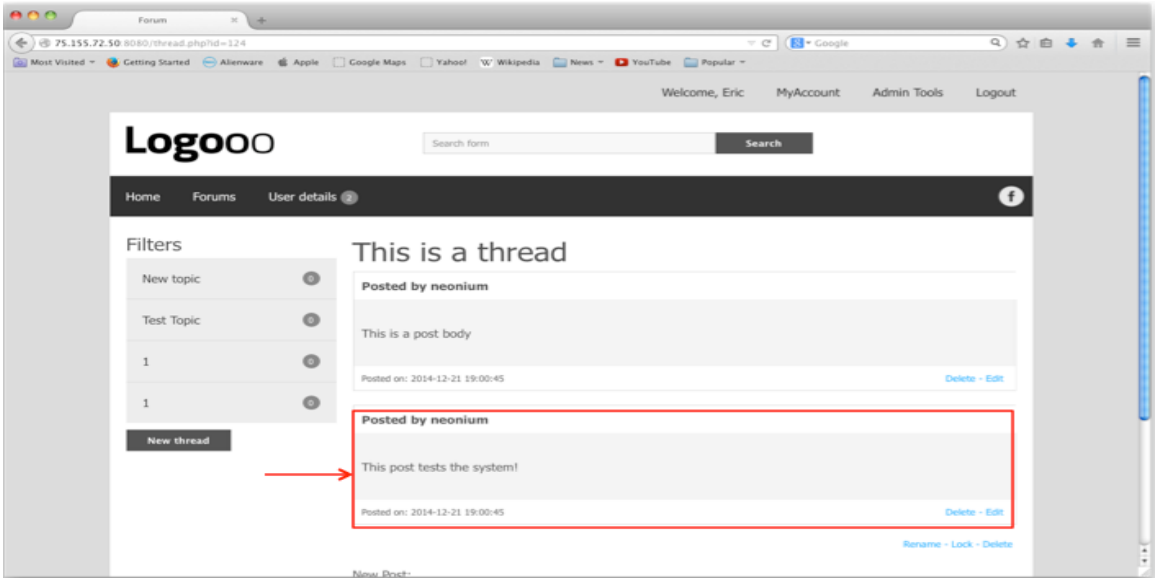

## 5.6 Writing a Post

A user can comment a given thread by adding a post to it. First, one must select the appropriate thread (such as the “This is a thread” thread in the example below). Second, one enters the text the appropriate field marked by the first arrow, followed by a click on the post button.




## 5.7 Viewing the Update

Once the user click on the post button, the new post appears in the thread, highlighted below.



# 6. Known Bugs List

- When deleting a topic subtopics will not automatically be deleted. They will be orphaned and must be deleted manually.
- When entering input certain characters will cause the input to cut off at that locations. Known character include “ and ‘.