# Mike and Chris’s Super Forum

### Mike Fortner, Chris Newman

### http://web.engr.oregonstate.edu/~fortnerm/CS-275-Project/

### 1. Introduction

Our project is a general web forum system. A web forum is an online discussion board where users can post messages, pose questions, and create new topics that users may discuss. In this particular case, the forum will be used to seek help with programming and post ideas and questions.

Our users will primarily consist of college and high school seeking help with their programming classes.

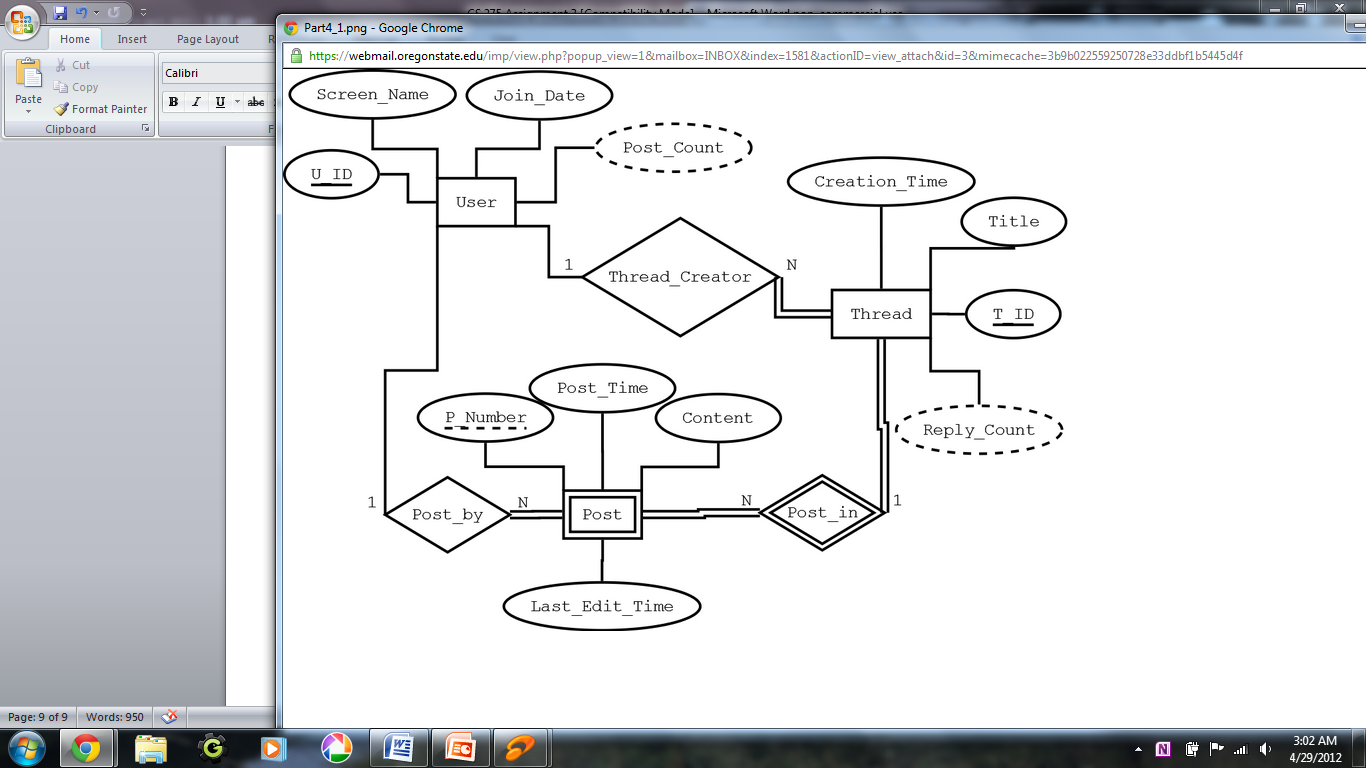
### 2. Detailed Application Requirements

Our web forum must:

* Have a simple, easy to use interface. The forum should be usable to those who have no experience with web forums.
* Provide basic forum functionality.
  + Users will be able to create threads, which are topics of discussion.
  + Users will be able to create posts under threads. Posts are individual messages that belong to a certain thread/topic.
  + The posts and threads will be displayed in the form of a simple GUI.
* Users will be able to create individual user accounts based on their personal information.
  + Each user’s post will be assigned to each respective user. For instance, if Chris creates a post, the post belongs to Chris.
  + Threads also belong to the respective users who created them.
* The forum can be viewed by anyone. However, the forum can only be edited by individuals with user accounts

### 3. Design

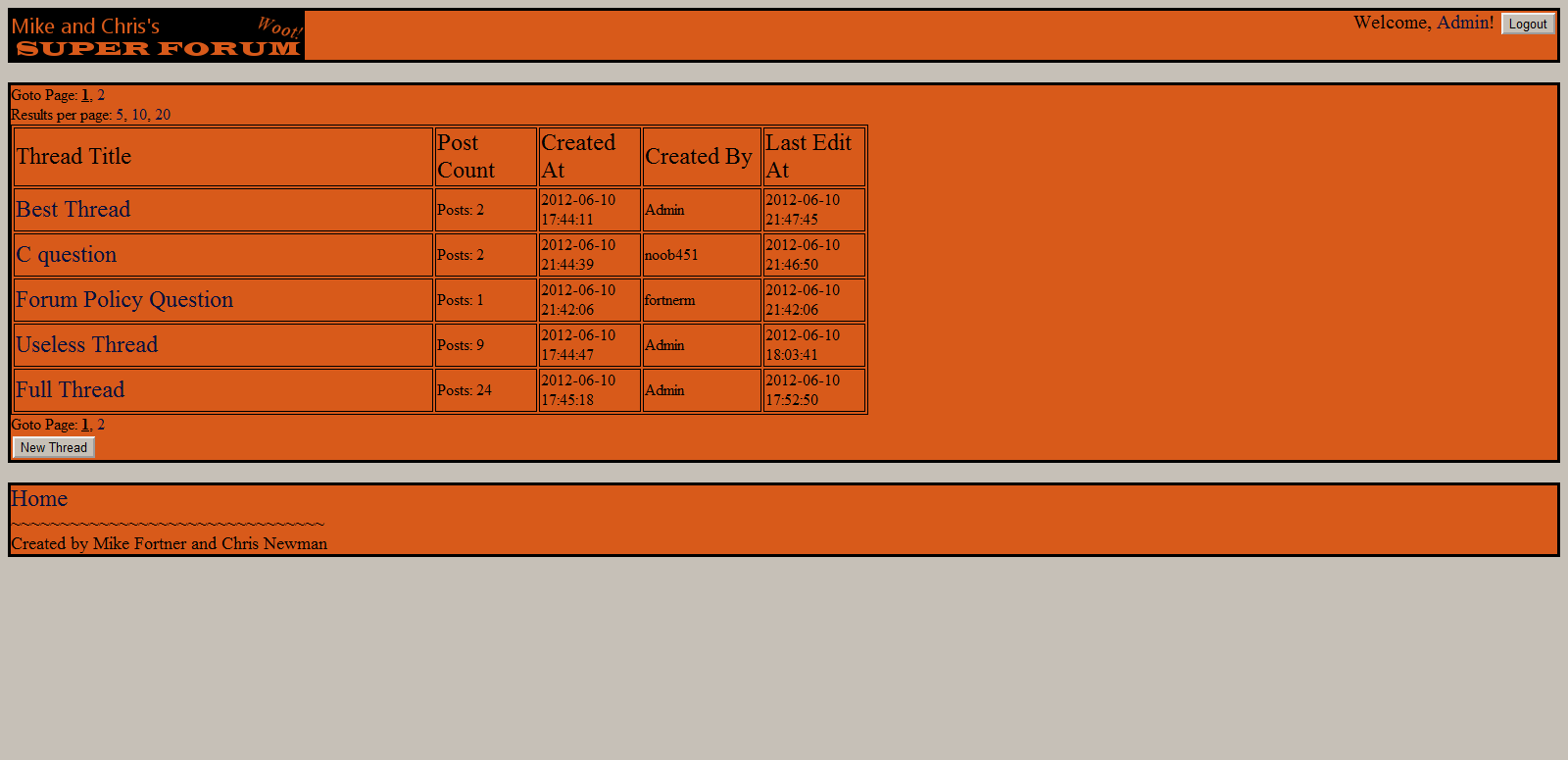
ER Diagram:



Our database consists of three types of entities: Users, threads, and posts. Post must belong to the users who created them, and the threads they were created in. Threads belong to the users who created them.

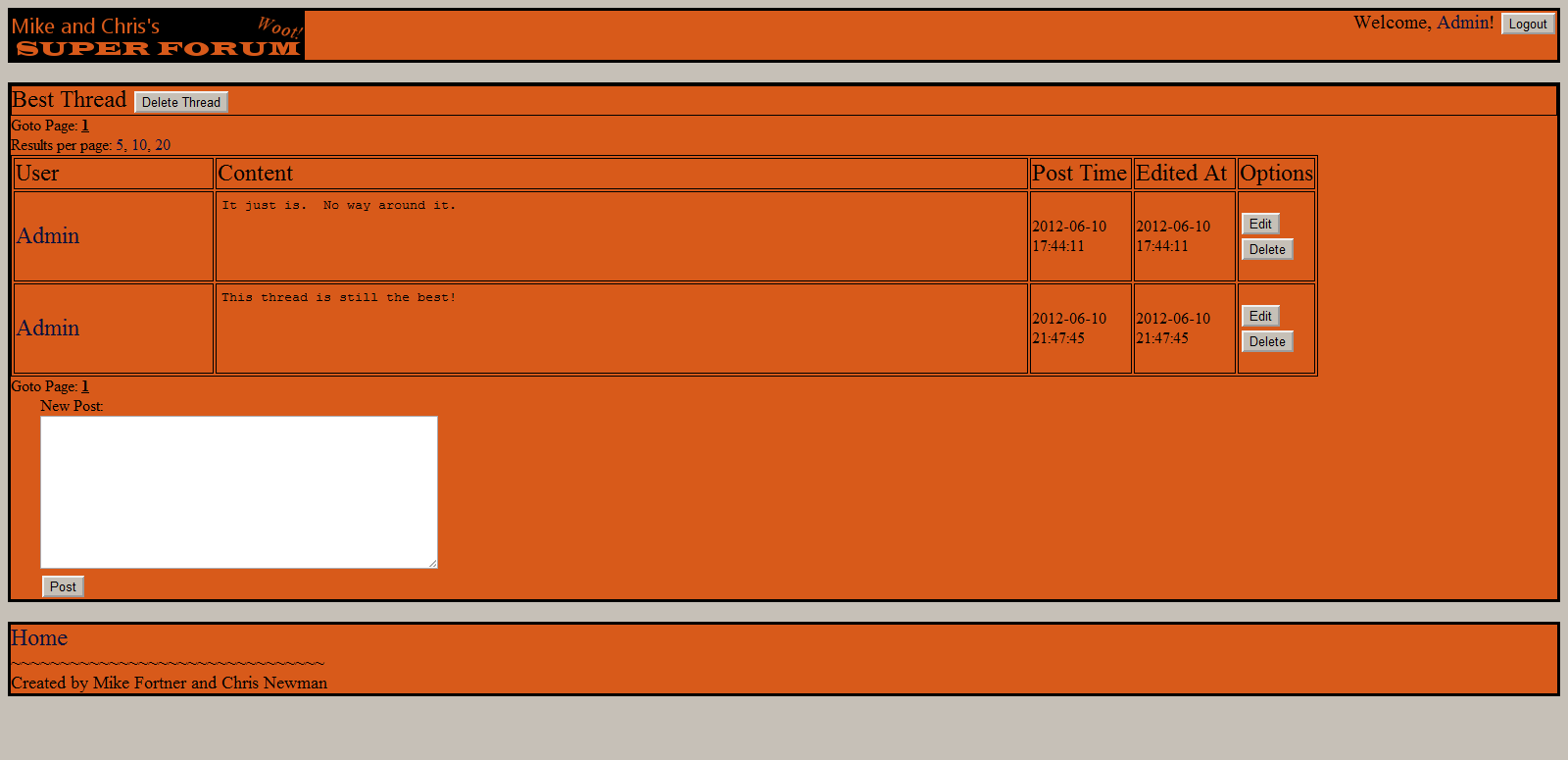
Our database can be accessed and edited from our website.

Homepage:



Each thread can be accessed from the table in the middle. Each thread’s number of posts and last post time are displayed. Users and guests can view threads by clicking on one of the thread titles. Only users can edit threads.

Thread Page:



Each thread contains a table with each post. Each post displays the name of the user who created the post, the content of the post, the time the post was created and last edited, and options to edit and delete the post. Users (who have created an account) can create new posts in the text field at the bottom. Guests without an account will not see this text box.

If a guest wishes to create threads and posts, they may create a new account by clicking the link at the top right of the screen.

New user account page:

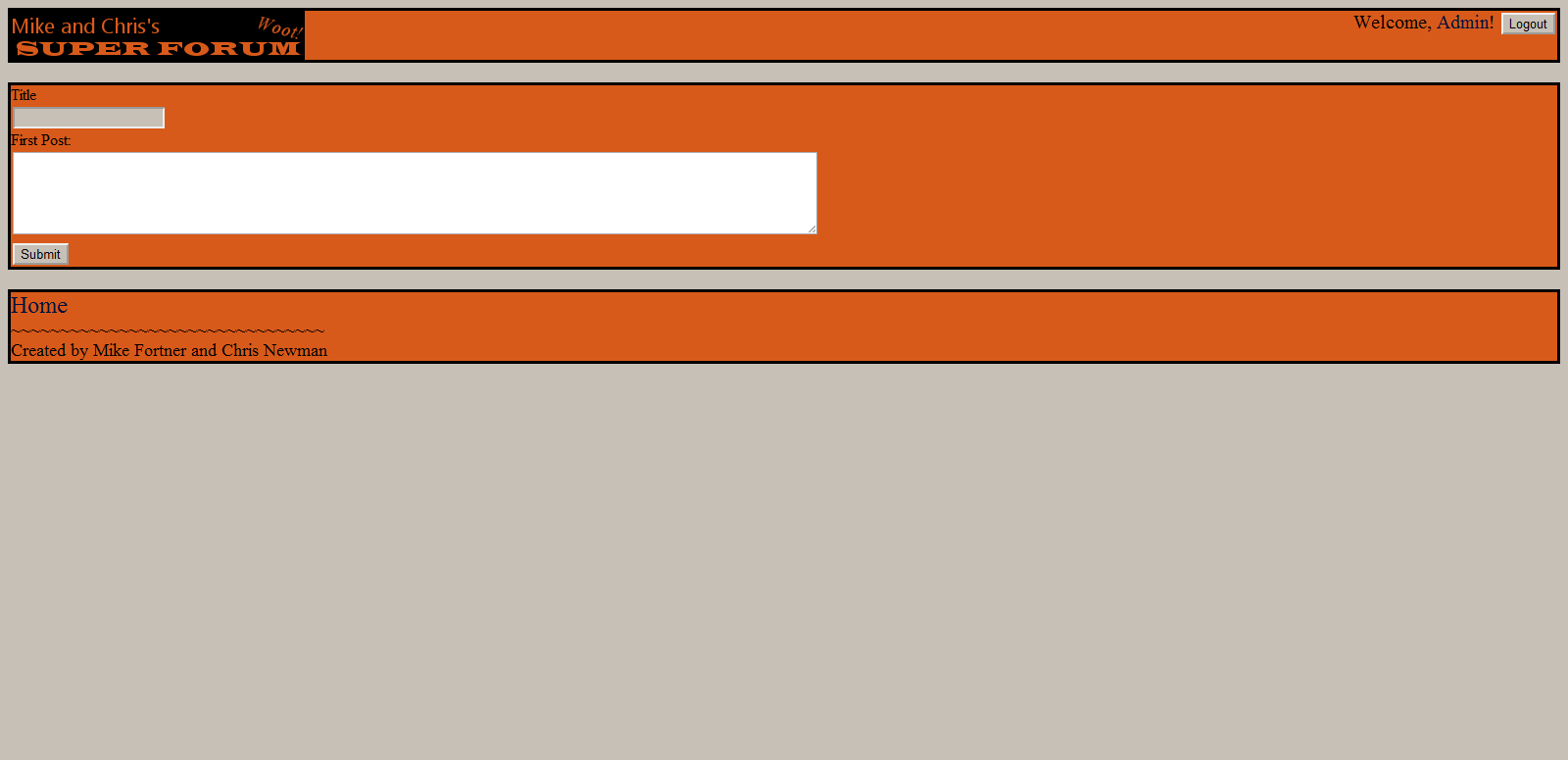


Here, guests can enter their desired username and password. They are automatically assigned a user ID so they can be identified by the database. Once a guest has created an account, they gain user privileges such as editing and creating posts and threads.

Our website is stored on Oregon State’s Engineering website. Only the database administrator has full control over the website and database. Users can edit and create threads and posts, and guests can only view.

For security reasons, passwords are encrypted before being sent to the database. This prevents user’s passwords from being stolen and used on other sites they may have repeated them on.

New Thread Page:



A simple page where a user enters a title for their thread and writes the first post.

### 4. Implementation Details

Much of our web forum’s functionality relies on the use of a DBMS. The forum must store data for user accounts, threads and posts and be able to access data at any time. For users to have their own personal accounts and to create threads and posts, their username and password must be stored on a database.

Specifically, here are the application requirements that require a DBMS:

Our web forum must:

* Provide basic forum functionality.
  + Users will be able to create threads, and store them in a database
  + Users will be able to create posts under threads. The posts are linked to specific threads using foreign keys.
* Users will be able to create individual user accounts based on their personal information.
  + Each user’s post will be assigned to each respective user via foreign keys. Usernames and passwords are stored in the database, and passwords are encrypted. Threads also belong to the respective users who created them.

Example queries:

1)

'SELECT Thread.T\_ID, Title, Creation\_Time, COUNT(\*), Screen\_Name, MAX(Post.Last\_Edit\_Time)

FROM Thread, Post ,User

WHERE Thread.T\_ID=Post.T\_ID AND Thread.U\_ID=User.U\_ID

GROUP BY Thread.T\_ID

ORDER BY MAX(Post.Last\_Edit\_Time) DESC

LIMIT '.(($curpage-1)\*$per\_page).','.$per\_page

This query gets a list of rows for the table in the homepage. The values of $curpage and $per\_page control the pagination of the content, so only the entries needed for the current page are retrieved from the database.

2)

SELECT User.U\_ID, P\_Number, Screen\_Name, Content, Post\_Time, Last\_Edit\_Time

FROM Post NATURAL JOIN User

WHERE T\_ID='.$T\_ID.'

ORDER BY Post\_Time ASC

LIMIT '.(($curpage-1)\*$per\_page).','.$per\_page

This one does the same thing, but for an individual thread.

3)

SELECT Screen\_Name

FROM User

WHERE Screen\_Name LIKE "'. $username.'"

Used to check if a username is already in use in the database. Is left case insensitive to avoid confusion between similar names.