# CS374 – Intro to Database Management

# Database Project

# Rubric for Final Project

## Group Member #1: Bishesh Tuladhar

## Group Member #2: Phuc Cai

## Group Member #3: Xavier Betancourt

## Group Member #4: Nhat Ngyugen

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Requirements | Points | Awarded |
| Database Design | * ER/UML diagram included * Database conforms to ER/UML diagram * Keys and Foreign Keys are defined appropriately * Database redundancies are eliminated or documented | 30 |  |
| Queries | * Queries execute correctly * Queries in English included * Queries satisfy the client requirements | 30 |  |
| Application | * Code found easily (This is a department assessment project) * Application works * Embedded SQL written as appropriate * Code well thought out, well commented * Application is well tested – that is, it is not easy to break | 40 |  |

**Whitbook Final Project Details**

Whitgit: <https://whitgit.whitworth.edu/pcai22/cs374Final>

**Database**

Hostname: api.hphucs.me

Username: cs300

Password: Whitworth000

**Overview:**

A social media app where users will have their own personal account, will be able to make short posts, have a following system, like and dislike, and will be able to read other users’ posts and save them. We are thinking about building a micro-blogging system that functions like Twitter. That is, having a character limit for posts.

The application would have a GUI that allows the user to interact with the app. The user would have the option to create an account or to login into an existing one. From there, the user can start writing posts and they would be saved in the app. There would be multiple tables holding all the information that the user is creating in the app. A basic foundation we are looking for to build is:

**Schemas**

CREDENTIAL(UserName, ID, Password, Email)

USER (ID, UserName, FName, Lname, Gender, Dbirth, DateCreated)

POST(PostID, Content)

ACTION(ActionID, UserID, PostID, Time, ActionDescription) //like, create post, save post

SAVED\_POSTS(PostID, UserID)

FOLLOWEE(UserID, FolloweeID)

**Project Management Schedule**

Who is doing what by when

Week 1 : ER and UML Diagram and making the base database (Together)

Week 2 :Start developing the base app that will use the database. (Phuc - Xavi)

  Start working on the necessary queries that the app will use (NhatMinh - Bishesh)

Week 3 :Finish the base app (that includes user interface and functionalities).

 Populate the database with incoming information from the user of the app. Also, test out            the database with multiple queries.

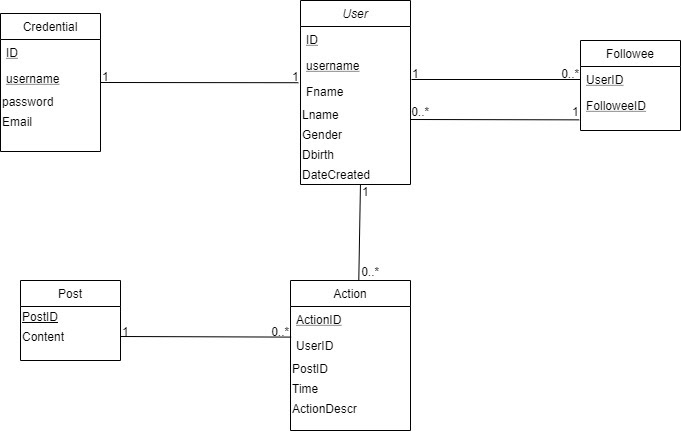
Week 4 : Debugging the app and finishing up details for the app.

   Working on a demo presentation for the class

   Make sure the database is well set up for the population of data.

**Logical Diagram**

Logical diagram in UML or E-R



Discussion of how your data model will satisfy the needs of your application

* The app will have the login information that will store the credentials of the users in the back. The users will be able to see the name, gender, birthday and date created of the other people that they follow. The user can create posts, like, dislike, and follow through the action table. The list of followers will be shown to users so that they know who they follow and who follow them, this is done through the followee table.

Discussion of alternative designs that you did not do (and why)

* We chose to do the UML instead of E-R because it is clearer to what we are doing and what we need. Like we have the tables and the data that we need then we know if there is how many people, either 0, 1 or infinity is affecting the table.

**Queries**

* Create account: input: username, password, email, the ID is auto generated
* Create post
* Following someone
* Like a post
* Check credential: input: username and password, the function will check if the input is the same as the one stored in the database, output true/false.
* Check followers: input: your user id, output: how many followers you have.
* Search post: input your user id, output: how many posts you have.
* Get Followers list: input: your user id, output: list of information of all the followers.
  + Join 2 tables follower and user.
* Get Follower post
* Delete follower

English description of query

1. Create account: Add an entry to the user table with all their information, add another entry to the credential table with the username, userID, password, email.
   * Entities : Username, Password, Email, ID, Fname, Lname, Gender, Dbirth, DateCreated
   * Relationship : User and Credential
2. Retrieve account/check credential: Retrieve all the information from the credential table based on the email that the user put in.
   1. Entities: ID, Username, Password, Email
3. Create post: Add a post to the post table with all the contents.
   * Entities: PostID, Content, ActionID, UserID,Time, ActionDesc
   * Relationship :  Post and Action
4. Save a post: Create an action for saving a post. This will use the last instance of the action table, ActionDescription = saved = “true”)

* Entities:  ActionID, UserID, PostID, Time, ActionDesc
* Relationship: Action, Saved\_posts

1. Count Followers : Returns the total number of followers a username has.

* Entities : UserID, Fname, Lname, Gender, Dbirth, DateCreated, FolloweeID
* Relationship : User and Followee

1. Like/dislike a post: Create an action for liking a post. Add an entry to the action table, with the ActionID auto increment, userID is the ID of the user who likes the post, postID is the ID of the post that the user likes, Time is the time when the user likes, ActionID is the ID for the action like.

* Entities: Action: ActionID, userID, postID, Time, ActionDescription
* Relationship: Action

1. Get follower post : Finds the post of a specific followee user

* Entities : Post: Id, userID, Followee : userID, authorID
* Relationship: Post and Followee

1. Delete follower: Deletes the specific followee from the list

* Entities : FolloweeID
* Relationship : User and Followee