Stage 6: Project Report and Demo Video

For this stage, your team is expected to submit the final project deliverables, including a project reflection report and a video demo.

Your project reflection report should contain the following topics. The report would be graded by completeness and correctness.

1. Please list out changes in directions of your project if the final project is different from your original proposal (based on your stage 1 proposal submission).
   a. Relational database schema was changed.

      (1) Since we cannot store an array in each row of the Group table, we deleted the Group table and created a relationship table ”UserPost” to specify the group.

      (2) We deleted the Purchase table and added the Post table, Category table, Payment table, and UserProduct table.

   b. We didn’t use Google/Twillio login API to generate an account string identifier. Instead, we let users input their own information to register and login to our application.
   c. For the creative component part, we implemented a UIUC video because our target users are UIUC students and alumni.
   d. UI was changed to a better version compared to stage 1.

2. Discuss what you think your application achieved or failed to achieve regarding its usefulness.

   - Achieved
     1. CRUD functions for posts
     2. Stored Procedure to analyze and categorize users based on their engagement
     3. Login and Register pages
     4. Clean User Interface
   - Failed
     1. Input: We still required users to input their userID/postId while updating and creating data.
     2. Token: We don’t have token authentication and expiration for now.

3. Discuss if you changed the schema or source of the data for your application

   - We changed our database schema because we found that relational databases could not save array data in a column. The detailed changes for a relational schema can be found in point 1.

4. Discuss what you change to your ER diagram and/or your table implementations. What are some differences between the original design and the final design? Why? What do you think is a more suitable design?

    - We originally have a Group entity to store the information of a group and a Leader entity to store each post initiator. For our final design, we replaced the Group entity with the UserPost relationship table, removed the Leader entity and put the group initiator information in the Post entity. In this way, we can resolve the redundancy of the Leader table and use the relationship table of UserPost instead. Also, we created another relationship table UserProduct to store the information of the products bought by each user. Lastly, we added another AnalyzeTable to store our analysis information.

5. Discuss what functionalities you added or removed. Why?

    - We added the functionality of the “User Analysis” function. We use a stored procedure to loop over the user table and categorize them into 5 sections based on the number of posts they have posted on our application. Also, we want to know what product category is the most frequently posted for each user. Besides meeting the requirements, we believe it is important to analyze our users to further improve our application or increase our base.

6. Explain how you think your advanced database programs complement your application.

    - Understanding the user’s usage pattern and their engagement on the website would help us identify the most active and not active users. This information would help us send some advertisements for those who don’t use our website much or improve our application based on their feedback in order to increase our user engagement. Thus, we believe our advanced database programs complement our application well.

7. Each team member should describe one technical challenge that the team encountered. This should be sufficiently detailed such that another future team could use this as helpful advice if they were to start a similar project or where to maintain your project.

    - We did not design a table to store the login token and user session, so we had a hard time figuring out how to let users know their userID in order to input some data.
    - When we are designing APIs to get post information, we would need multiple APIs to get the whole data we need. For example, we need one API to get post information such as post product, product price, and initiator. We need another API to get group information. If

we implement a NoSQL database such as MongoDB, we can incorporate these APIs into one query.
- Initially, our auto-generated data in the database doesn't make sense, so we need to revise all the data in different tables in order to make our SQL query results reasonable when we retrieve the data.
- When hosting the application on the GCP, we forgot to open TCP port 3001, 3002 on the GCP firewall.

8. Are there other things that changed comparing the final application with the original proposal?

   All the changes have been mentioned in point one. There are no other things that changed comparing the final application with the original proposal.

9. Describe future work that you think, other than the interface, that the application can improve on

   - Auto delete post when expiration date hits
   - Connect with 3rd party authentication
   - Embed product picture in the post
   - Create a chat room for users
   - Set up token session expiration date, and token authorization (Show token in console
   - Add notification when people join/ leave/ delete post
   - Add online payment
   - Auto fill the postID and userID for users
   - Personal info -> profile page

10. Describe the final division of labor and how well you managed teamwork.

    - User Login/Signup: Ray Chang
    - Home Page: Ken Wu
    - Search Post Function: Ken Wu
    - Create Post Function: Thomas Huang
    - Edit Post Function: Ray Chang
    - Join Post Function: Thomas Huang
    - User Analysis(store procedure): Meg Wu
    - Database: Meg Wu