CSCI 4370: Final Project: Fall 2023

In this project you will design a database and develop a dynamic web application for an interesting problem of your choice.

Requirements

The project should fulfill the following requirements:

- Start by writing down the following as the initial step.
    - Title for the project
    - Describe the problem or the domain.
    - Describe the solution you develop. Include what web pages it is going to contain.
    - Preliminary ER diagram.
    - The technologies you will be using.
    - Submit this with the final submission.
- Database design:
    - First, you should use the **ER model** to capture the data aspects for your application.
    - The ER model should have **at least 10 meaningful entities** and it should capture relationships among them.
    - Then, you should apply '**ER to table' conversion** rules to get the schema for the database with appropriate constraints.
    - You should **normalize the resulting tables** into BCNF or 3NF normal forms. In order to do this, you should write down the **functional dependencies** for your application.
    - After that, you should use the **docker based MySQL instance** we use in the class to create your database schema.
- The web app should have **at least five meaningful dynamic web pages** that interact with your database.
    - The server side of your Web app should be implemented using Java using a **maven project structure** like we discussed in the class.
    - A **JDBC connection** like we discussed in the beginning of the class should be used to interact with the database from your web app.
- Query requirements:
    - You should use at least **10 independent meaningful SQL statements** that perform data retrieval, insertion, deletion and update. Each query should be connected to the pages of your web application in a meaningful way.
        - The queries should also include aggregation and join statements with sufficient complexity.
    - You should use **SQL prepared statements** when running SQL statements from your application mainly to reduce the risk of SQL injections.

- Indexing:
    - You must use at least two indexes to optimize the performance of at least two of your queries.
- User signup and login:
    - Your application must support a signup and login process for users.
    - You should make sure to use **secure approaches** to store user passwords in the database.
- Data requirements:
    - The data you use for the application must mainly come from a realworld source.
    - There should be at least one table with more than 1000 rows.
    - The database of the application should be **populated with meaningful data** to be able to demonstrate the app.

You are allowed to use external libraries for both server side and client side as long as you check and abide by the licensing restrictions and get **prior permission** from the instructor to use them and include them in the readme file of your submission.

Submission

Your submission should have the following content.

- The PDF report you created as the initial step with name **prelim.pdf**.
- A PDF report with the name **db_design.pdf**
    - Your ER model
    - Resulting relations after converting the ER model to relations
    - Functional dependencies that you have identified
    - Normalization steps and final normalized relations.
- Working **code for the web app**.
- DDL queries to create the database schema for the web app in a file with the name **ddl.sql**
- All queries you used in the web app (both data retrieval and data modification) in a file with the name **queries.sql**
    - Include a brief comment above each query explaining the purpose of the query in the context of your web app. Also provide the URL path of the page that uses it in the web app.
- Data to demonstrate the app as SQL insertions in a file with the name **data.sql**
- **datasource.txt** that describes the source of your data and mentions what data parsing and preparation mechanisms were used.
- **perf.txt** file that lists the statements you used to create indexes:
    - This file should also include the queries that were affected by the indexes.
    - Under each query provide time consumption before and after adding the index. This should be done after populating the database.
- **security.txt** file that describes the approach you used to securely store user passwords.
- A **demo video** of the web app that includes how to compile and start it.

    - The demo should show each web page you developed with a brief narration.
    - The database of the app should be already populated when you do the demo.
    - Make sure to showcase **all requirements that you fulfilled** within the demo.
    - Each team member should participate in the demo presentation. Mention your name before starting to talk.
    - Suggestion: You can do this by making a recorded Zoom call with your team members.
- A **readme.txt** file with team member names and group name in a readme file.
    - List what each team member did under their name.
    - Include all the **technologies** you used including 3rd party libraries.
    - Include the **database name**, **database username and the password** you used for the JDBC connection.
    - Provide three **test username and password pairs** from your demo data.

The work should be done in the groups you created on Piazza. Only the **group representative** should submit the assignment on **eLC**. Create a zip file of your working code with other required files and submit it on eLC.

Due date: Fri, Dec 08, 2023 @ 06:30 PM (this is the official final exam end time for the class).

This is a hard deadline. No late submissions are accepted and no extensions will be possible because this is the very end of the semester.

Important Notes:

- Projects that do not compile or run can not be graded.
- Please redownload and check your zip file after the submission on eLC to verify that no files are corrupted. We will not be able to grade your project if the files are corrupted.
- You may lose points if your submission is not complete and has missing deliverables.

Please remember the **academic honesty policy** for the course. You are not allowed to publish or store the assignments or the solutions in places accessible to others.

Note: If you have any questions, please post on Piazza (link on eLC).

Important Point Allocations

The grade will be given out of 100 pts.

- 10pts: Data Set Quality:
    - A significant amount of data you use should be realworld data.
    - You should have used automated data parsing or other preparation mechanisms.

    - Dataset should be large enough to take advantage of indexing.
- 5pts: App functionality:
    - The application features should not be toy-like.
    - The application should be complete and should not have glitches.
- 10pts: Technical Depth of queries and the database design:
    - Queries and the database should demonstrate your proficiency.
    - The database side should not look toy-like.
- The rest of the points will be allocated to fulfilling the requirements.