NYU Tandon School of Engineering
Computer Science and Engineering
CS6083, Spring 2024

April 27, 2024

Project #2 (due May 14, 2024)

Overview

For the second part of your project, you are required to develop a web-based user interface for the relational database schema designed in Project 1. The interface will facilitate user interactions within a local community platform, allowing residents of specific neighborhoods and blocks to communicate and engage in discussions effectively. Use PostgreSQL as your database system.

Functional Requirements

1. User Registration and Authentication: Allow new users to register and existing users to log in. Registration may involve specifying home addresses or selecting a location on a map, as per the choices implemented in Project 1.
2. Profile Management: Users should be able to create and edit their profiles.
3. Communication and Interaction:
    - Enable users to post messages, reply to threads, and interact within their designated blocks or neighborhoods.
    - Users should be able to apply to join a block, gaining membership upon approval by sufficient existing members.
4. Information Display and Updates: Provide users with access to different views of the messages and discussions relevant to them:
    - A main feed page showing all recent threads, categorized into neighbor feeds, friend feeds, block feeds, and hood feeds.
    - Options to view threads with new messages since the last visit, and profiles of newly joined members of their blocks or hoods.
5. Navigation and Search Functions:
    - Implement search functionalities to find messages based on keywords or geographical proximity.
    - Users should have the ability to follow blocks outside their membership for read-only access to those threads.

6. Data Visualization (Optional): While not explicitly required, incorporating visual (mostl likely map-based) representations of data such as the density of interactions within a block or neighborhood could enhance user engagement.

Technical Specifications

- The interface should be accessible via standard web browsers without the need for public internet access. Typically, a web server on your local machine will suffice.
- You may use frameworks like PHP, Python Flask, or Django for backend development. It's crucial to integrate SQL queries directly, demonstrating your capability to interact with the database effectively during your demo.
- Ensure the interface is secure against common web threats such as SQL injection and cross-site scripting. Use techniques like prepared statements, input sanitization, and output encoding to protect the system.

Collaboration and Documentation

- If working in a group, all members must be familiar with all aspects of the design and participate in the demo.
- Document your design and implementation processes in a comprehensive report (15-20 pages). This should detail your architectural decisions, security measures, user guide, and any other relevant information. The submission of reports will be done on Gradescope. One member of the team can submit and add the other team member to it as a group submission.

Demo and Submission

- Schedule a demo with the TAs, choosing from several available dates around the semester's end.
- Consider implementing additional features for potential extra credit, subject to TA approval at the end of your demo.

Deductions

- Be aware that any crashes in the web application or database errors during the demonstration will result in deductions from the final score.

Additional Tips:

- Database Design: Know about your ER diagram and relational schema. Know your key constraints used and demonstrate space efficiency and normalization.

- SQL Queries and Database Population: Showcase SQL queries for specified tasks and ensure a meaningful population of test data. Demonstrate successful CRUD operations (create, read, update, and delete - the four basic operations of persistent storage), Measures against SQL injection and cross-site scripting attacks, and Support for concurrent access.
- Web-based UI Implementation: Implement functionalities such as user registration, login, and message posting. Create at least 4-5 different views (pages displaying the functionalities) for community interaction. Ensure measures against security threats and support for concurrent user access.
- Project Report: Provide a comprehensive and well-documented report. Include design explanations, ER diagrams, and sample queries. Log system sessions and prepare to show and explain the source code during the demo.
- Answers to Questions During Demo: Prepare to answer questions clearly and concisely during the demo. The demo slots will be 15 minutes long, so be precise.