Term Project
COP 4710, Spring 2025

Application: College Event Website

Problem:

Most universities in the country host events around campus and off campus. These events are organized by college students in most cases. Students are clustered RSOs (Registered Student Organizations) by different organizations, clubs, fraternities around campus. These events could be of different types: social, fundraising, tech talks, etc. Now, each university has a website where they post their events for the upcoming weeks. One needs to check the website to add each event to his/her calendar. These events are just official events and not all events around the university are included. Another limitation is that one has no way to track weekly events.

Project Description

You are asked to implement a web-based application that solves the problems. Any student (user) may register with the application to obtain a user ID and a password. There are three user levels: *super admin* who creates a profile for a university (name, location, description, number of students, pictures, etc.), *admin* who owns an RSO and may host events, and *student* who uses the application to look up information about the various events.

Admin can create events with name, event category, description, time, date, location, contact phone, and contact email address. A location should be set from a map (Bing, Google, open street map) with name, latitude, longitude, etc. To populate the database, one can use feeds (e.g., RSS, XML) from events.ucf.edu. Each *admin* is affiliated with one university, and one or more RSOs. A *user* can request to create a new RSO or to join an existing one. A new RSO can be created with at least 4 other students with the same email domain (university), e.g., @knights.ucf.edu; and one of them should be assigned as an administrator.

There are different types of events (social, fundraising, tech talks, etc.). Each event can be public, private, or an RSO event. *Public events* can be seen by everyone; *private events* can be seen by the students at the host university; and an *RSO events* can only be seen by members of the RSO. In addition, events can be created without an RSO (public events). Such events must be approved by the super admin. After an event has been published, users can add, remove, and edit their comments on the event, as well as rating the event on a scale of 1-5 (stars). The application should offer some social network integration, e.g., posting from the application to Facebook or Google.

When logged in, Student should be able to view all public events, private events at their university, and event of RSOs of which they are member. They will not be able to create events, but should be able to rate, comment and edit (update) their comments for any event.

To be provided:
To help students design/implement the above project properly, the following materials will be provided:

1. A discussion of candidate keys for entity Event.
2. A trigger to enforce the constraint "The activation of an RSO requires at least five members" will be provided in SQL. The code might need to be modified to work with the DBMS of your choice. An example of the trigger that works on MySQL (i.e., in PHP) is given.

Technical Requirements:

1. The design of the database must follow the design process: business operations/constraints, ER-model, the relational model, normalization, implementation, indexing, enforcing general constraints (e.g., triggers), etc.
2. The database must include at least 5 relational tables.
3. The application must have a browser-based interface and can be deployed on the Internet. Moreover, the application should have a professional-looking interface that is user-friendly.
4. The website and database must be able to support multiple concurrent users.
5. Sample data should be included, e.g., 10 uses, 5 RSOs, 20 events, 10 comments, 10 SQL queries…
6. Programming languages that can be used for the project: HTML, Javascript, PHP, Java, CSS, c#, and stored procedures. DBMS's: Oracle, SQL Server, and MySQL. Other languages and DBMSs are generally allowed provided that you must begin the design and implementation in SQL and must be able to modify the code in the platforms of your choice to implement required by the project.
7. All constraints must be enforced using the database's features that support those constraints, e.g., triggers.
8. Advanced features include, but not limited to, event feeds from the university's events system (e.g., from https://events.ucf.edu), social network integration, crash recovery policy/procedures, security, index-only/composite search-key indices, etc.

Grading:

- Demo 50%
- Report 50%

Deliverables:

- Due Dates: TBA
- Draft Design: ER diagram (example design might be given and modified for use), relational schemas, constraints enforcement (example code will be given and could be modified to implement in the database platform of your choice).
- Report: The design (ER diagram, relational schemas, …), database creation code (CREATE TABLE …, INSERT…, SELECT …, etc.), the software code (GUI, embedded SQL statements, etc.), screenshots of interface, output, comments/observations, lesson learned
- Demo: to the GTA over Zoom or in person.

- Submission: one compressed file *.zip via Webcourses containing all the files and codes and the report.

Note:

The instructor reserves the right to modify (add/drop) the technical requirements, due dates, grading weights, and to provide relevant information such as general constraint enforcement codes should it deems appropriate. All changes will be announced on Webcourses.