Group Report University Event Website

Section: COP4710-22 Spring 0001 Group Members:

Nawras Rawas Qalaji

Matthew Rodriguez

Michael Harris

Table of Contents:

Project Description PG) 1

GUI PG) 2

ER Diagram PG) 3

Relational Data Model and Sample Data PG) 3

SQL Examples PG) 4,5

Constraint Enforcement PG) 6

Advanced Features PG) 7

Conclusion PG) 7, 8

Project Description

This project involved creating a database and accompanying website to store and access information on different events hosted by various universities. These events can be separated into different categories, such as social or fundraising, and different types such as private, public and RSO only. A user may register and depending on their user level: super admin, admin, and student, may either create a university profile, create an RSO with minimum 4 other students and host events or simply search event information and post comments and ratings on those events; each user level has the abilities of the one below it. Students can comment on any events they have access to; RSO events are only viewable by their members, private events can only be viewed by the students attending the university hosting it and public events can be viewed by anyone. Anyone may request to create a public event however the request must be approved by a super admin. Admins may create private events and RSO events they are admins of, and super admins can create private and public events.

We assume that all users upon registration have the user level of student, and that whenever a user creates a RSO they automatically become an admin, however they cannot create events until their RSO becomes active. On creation an RSO is set to inactive until at least 4 other students join the RSO.

GUI

Software Used

* WAMP Server
* PhpMyAdmin
* MySQL
* Apache
* PHP
* HTML
* Localhost

GUI Example 1

Diagram

Description automatically generated with medium confidence

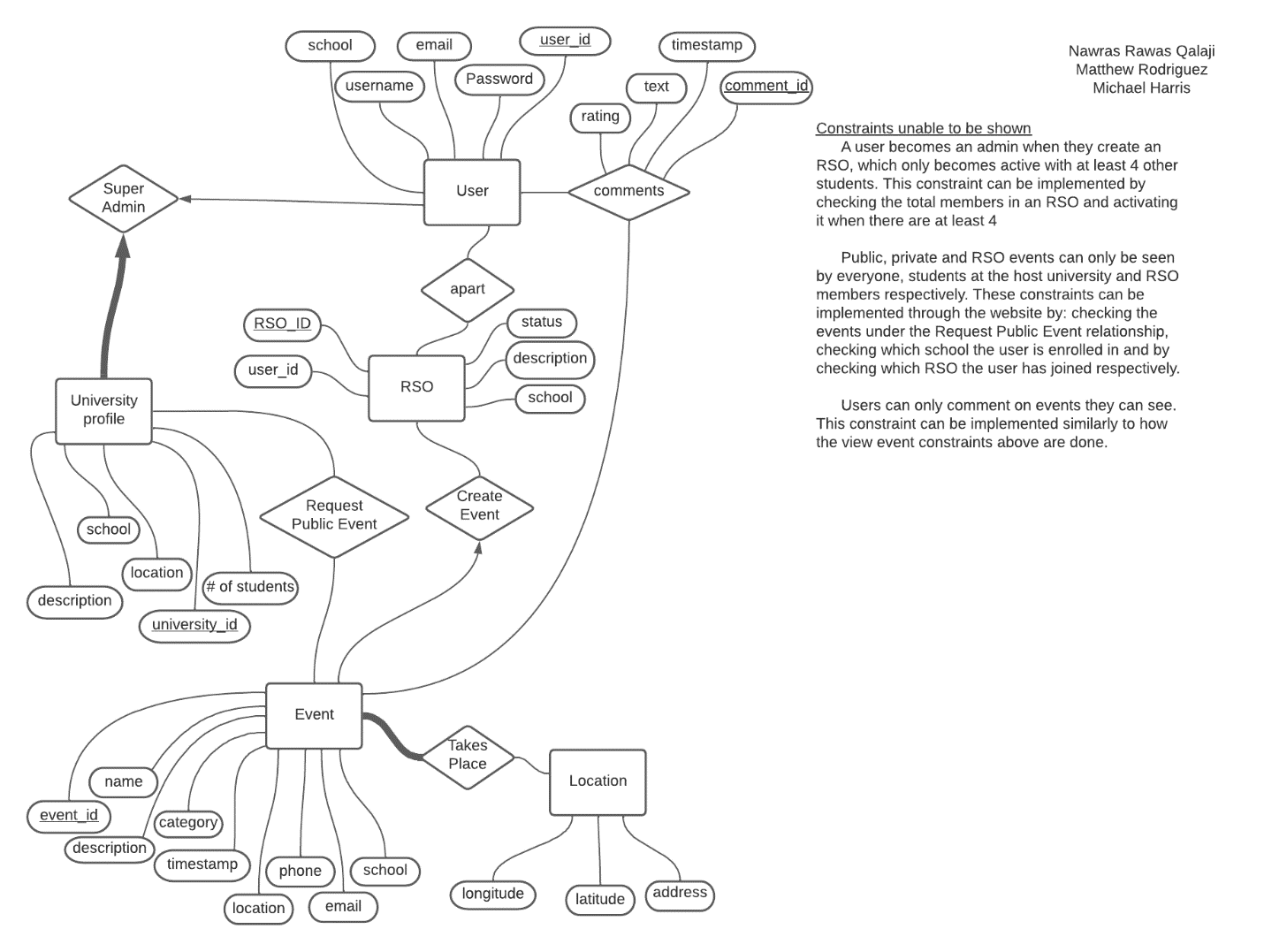
GUI Example 2



GUI Example 3

Graphical user interface, text

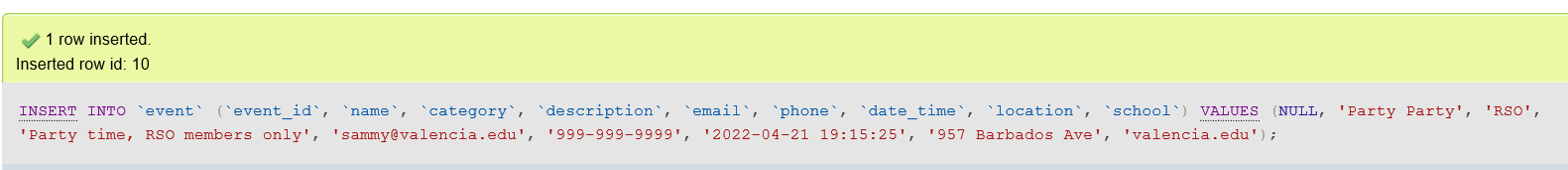
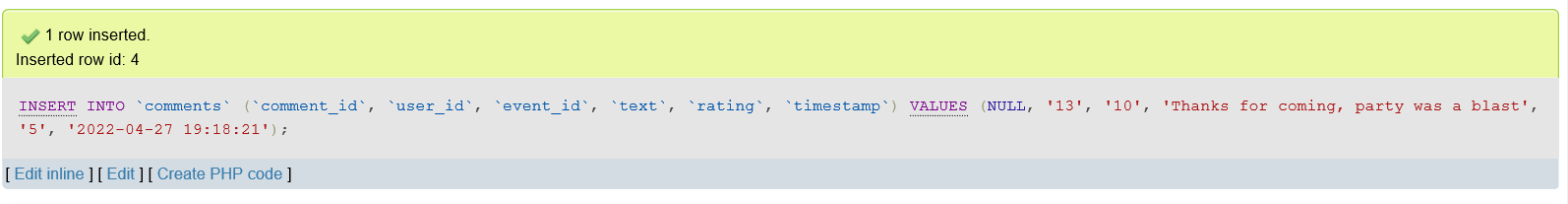
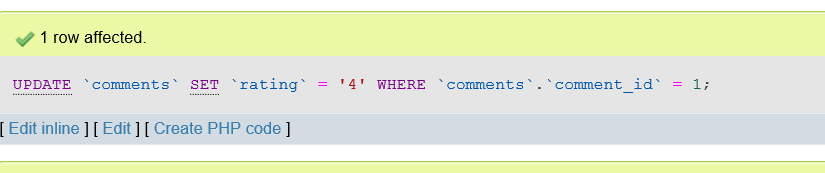
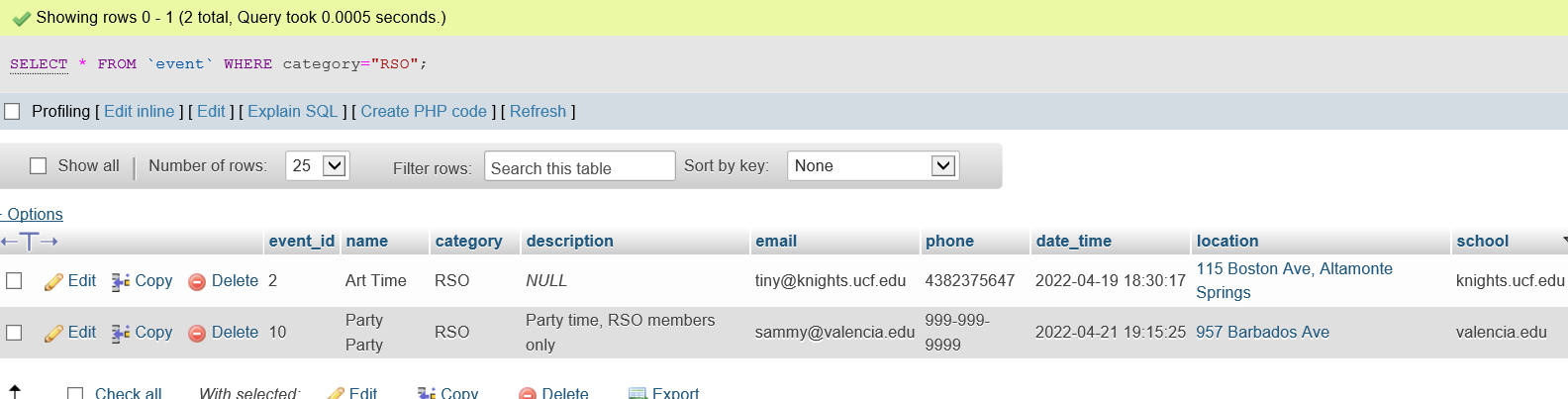
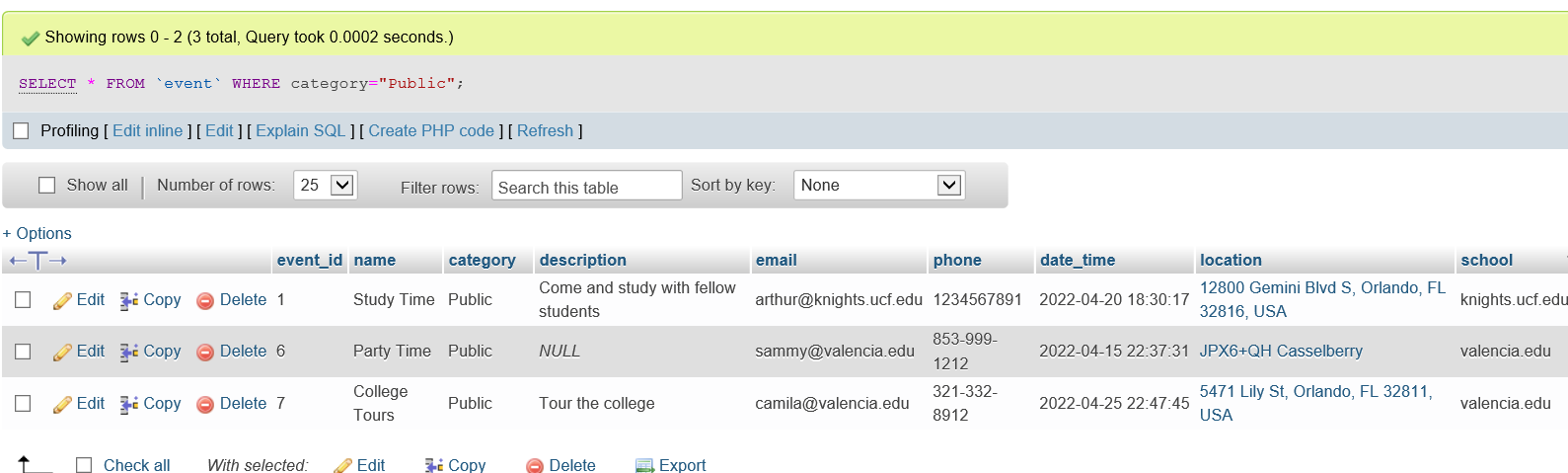
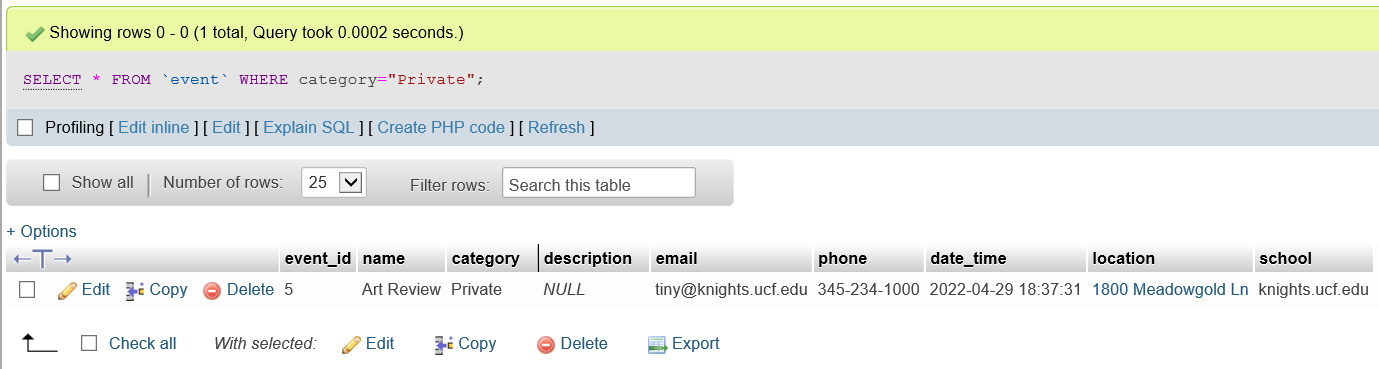
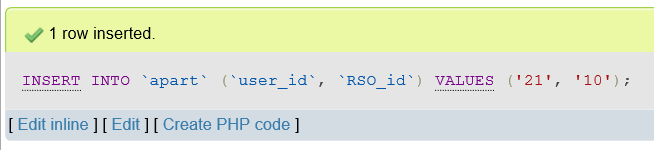
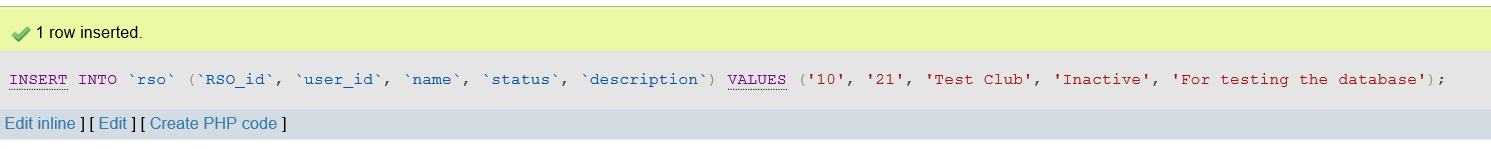
Description automatically generated

ER Diagram

Relational Data Model and Sample Data

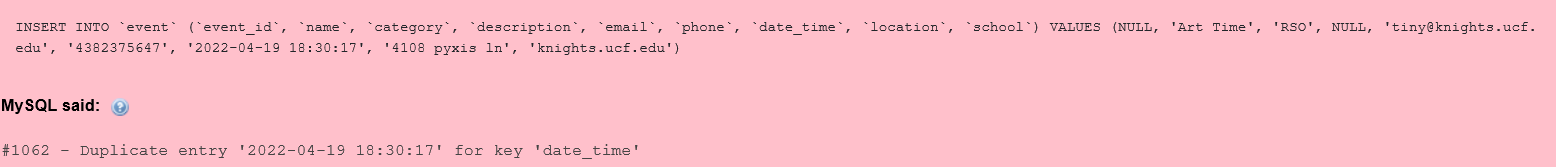
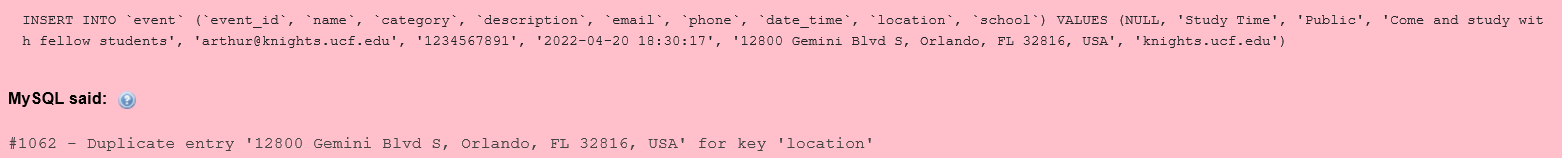
See included relationalDataModel.sql file

SQL Examples

see included files if pictures unclear

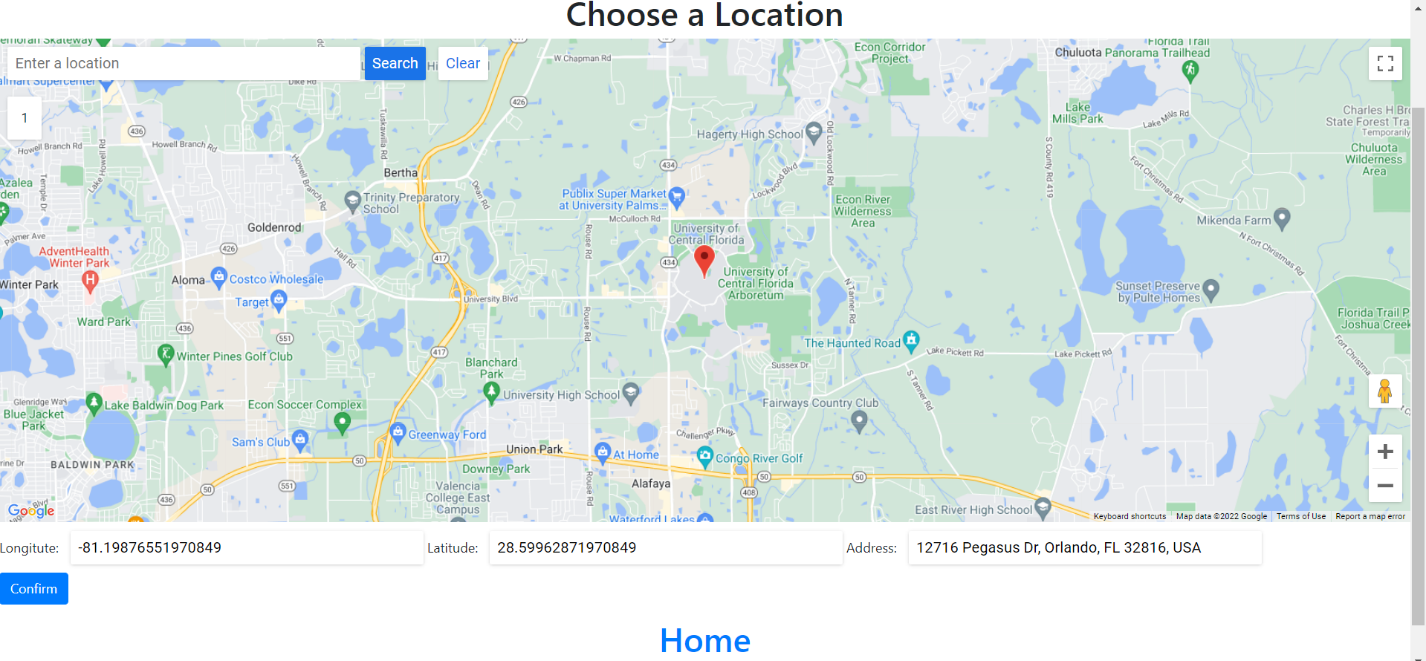
Constraint Enforcements

Also see attached files if pictures unclear



Advanced Features

The website uses google maps integration to allow the user to simply type in a location or place a pin on the google map to automatically fill in the information required for the event.



Conclusion

In conclusion our database was able to achieve the goals put forward. In terms of speed the database query response time is around 0.0002 seconds for each one. We mostly used primary indexing since our database did not have many attributes that would slow it down. We were also able to take advantage of the google maps integration which makes adding in locations much easier. One of the problems we encountered was first deciding how we wanted to setup the database. We ended up going through multiple iterations before we decided on one that would take advantage of both the constraint enforcement on the database side and website side. Another issue we faced was how we would make the RSO automatically change from active to inactive and vice versus. Initially we thought about simply using a trigger however after restructuring the database and learning more about web development we thought it might actually be easier to implement the constraint on the website side instead. However, we were pleasantly surprised to see that integrating google maps was easier than we expected due to how modular the external library was, it makes it much easier for software developers when we can take advantage of powerful resources easily.

This project has taught our team how to utilize phpMyAdmin more effectively and how to create a database and generate different queries. We also learned a great deal about web development and writing PHP in conjunction with html to query and display information from the database. Building a more advanced database would likely require better and more utilization of external libraries, updated versions of software and improving skills in PHP, MySQL and other general database management and API systems. A more advanced database might also require the use of triggers and assertions though for this project it felt like more work to implement these methods into the database than simply implementing them in PHP and HTML.