College Event Website Report

Khalil Oxborough
Spring 2023, COP4710-0001

Table of Contents

(3) Project Description

(5) GUI

(11) Entity Relationship Diagram

(12) Relational Model

(13) Sample Data

(16) SQL Examples

(19) Advanced Features

(20) Conclusion/Observations

Project Description

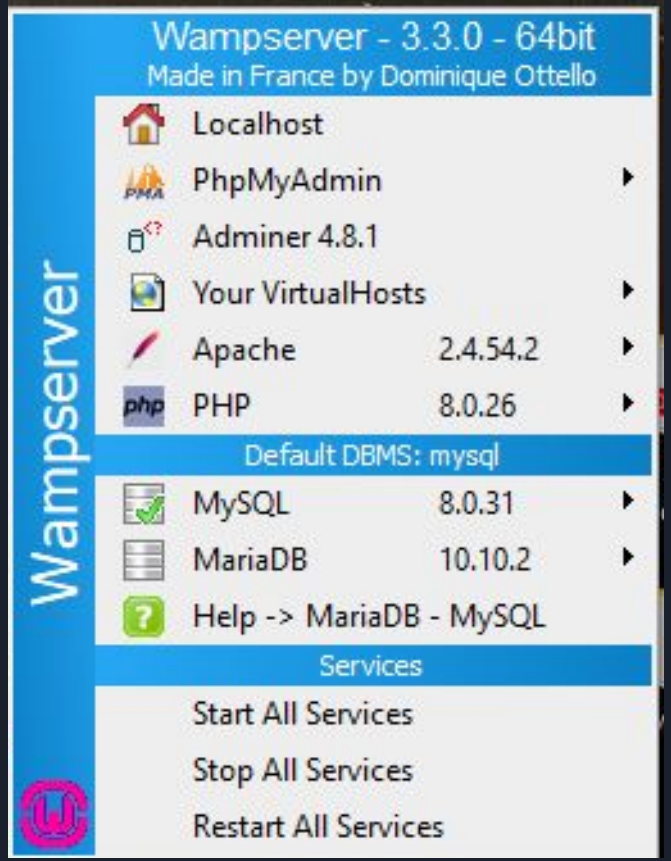
- You are asked to implement a web-based application that solves the problems. Any student (user) may register with the application to obtain a user ID and a password. There are three user levels: super admin who creates a profile for a university (name, location, description, number of students, pictures, 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.

Project Description (cont.)

- 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.

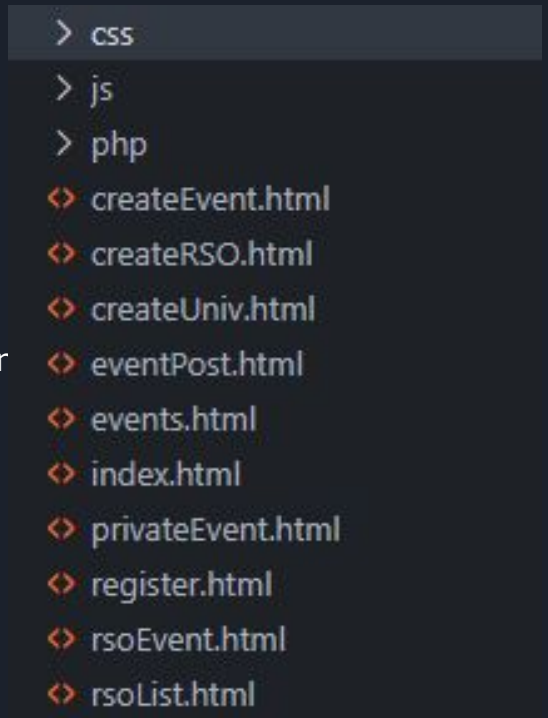
GUI (Platform)

I used a WAMP Server (Windows, Apache, MySQL, PHP) for the full project.



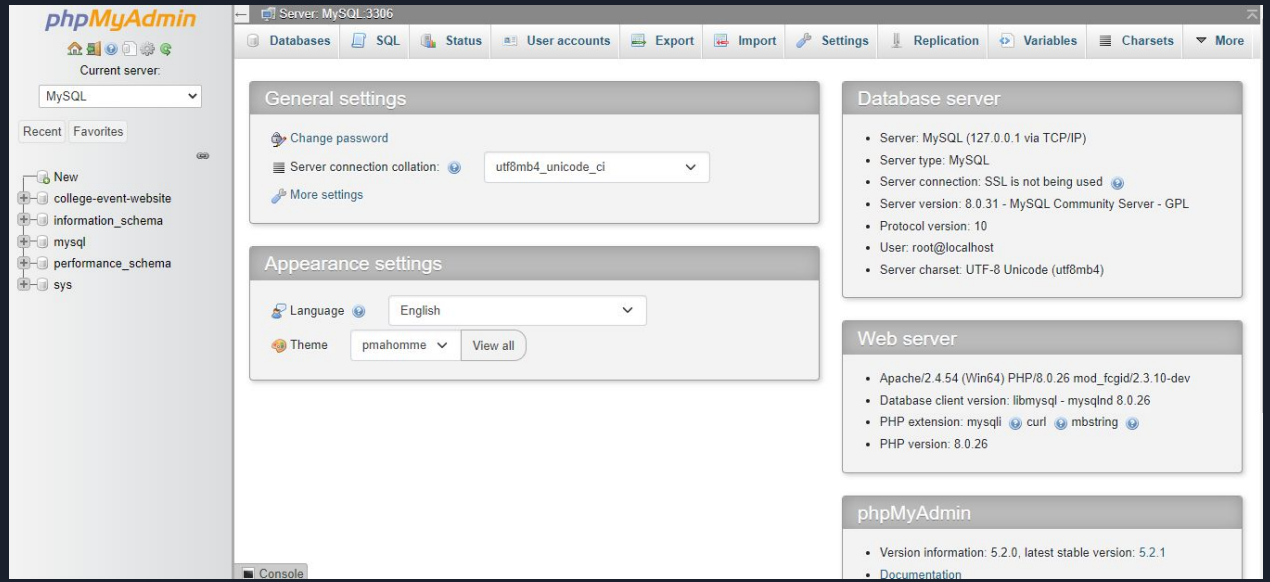
GUI (Languages)

My college event website used HTML and CSS for the structure and design of the website, JavaScript for the function of the website, PHP for the API, and SQL for queryir

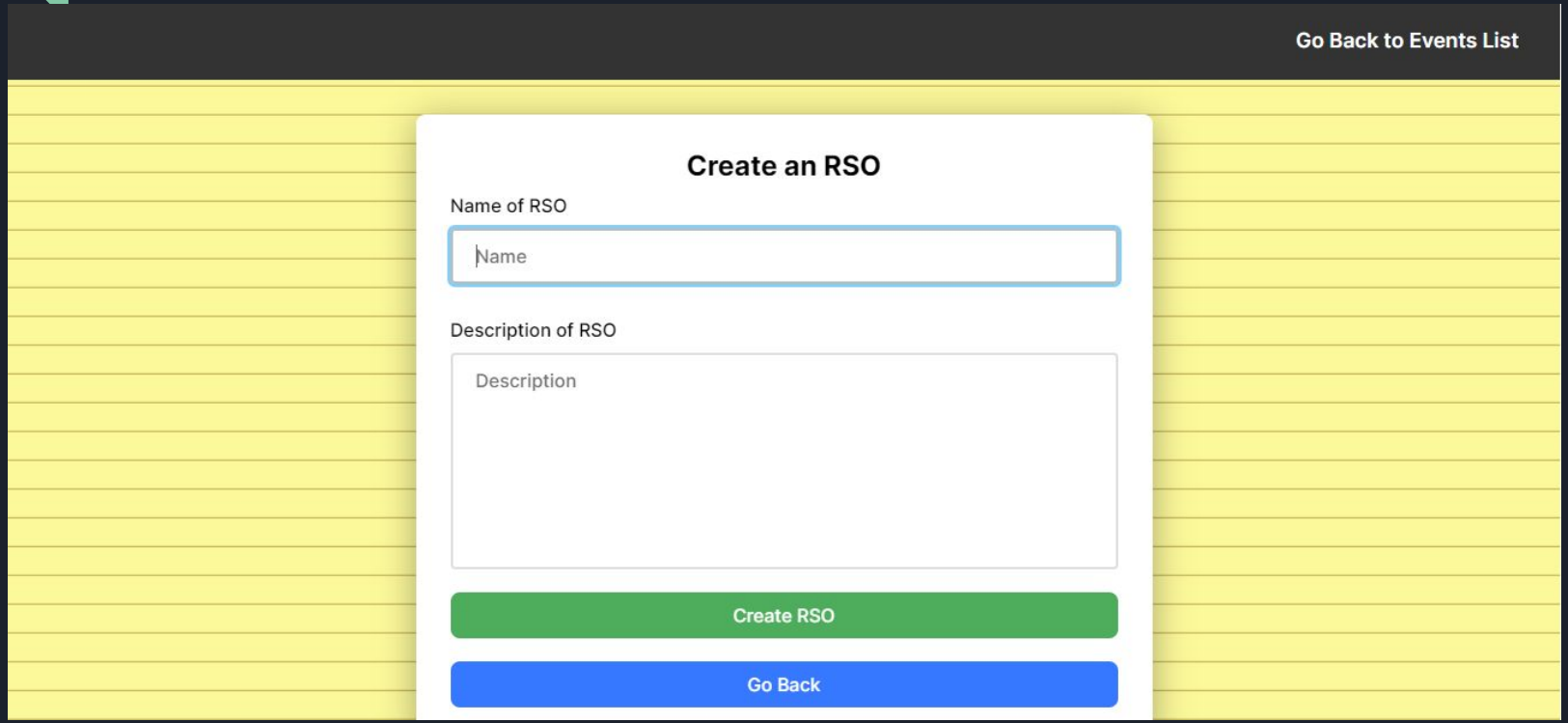


GUI (DBMS)

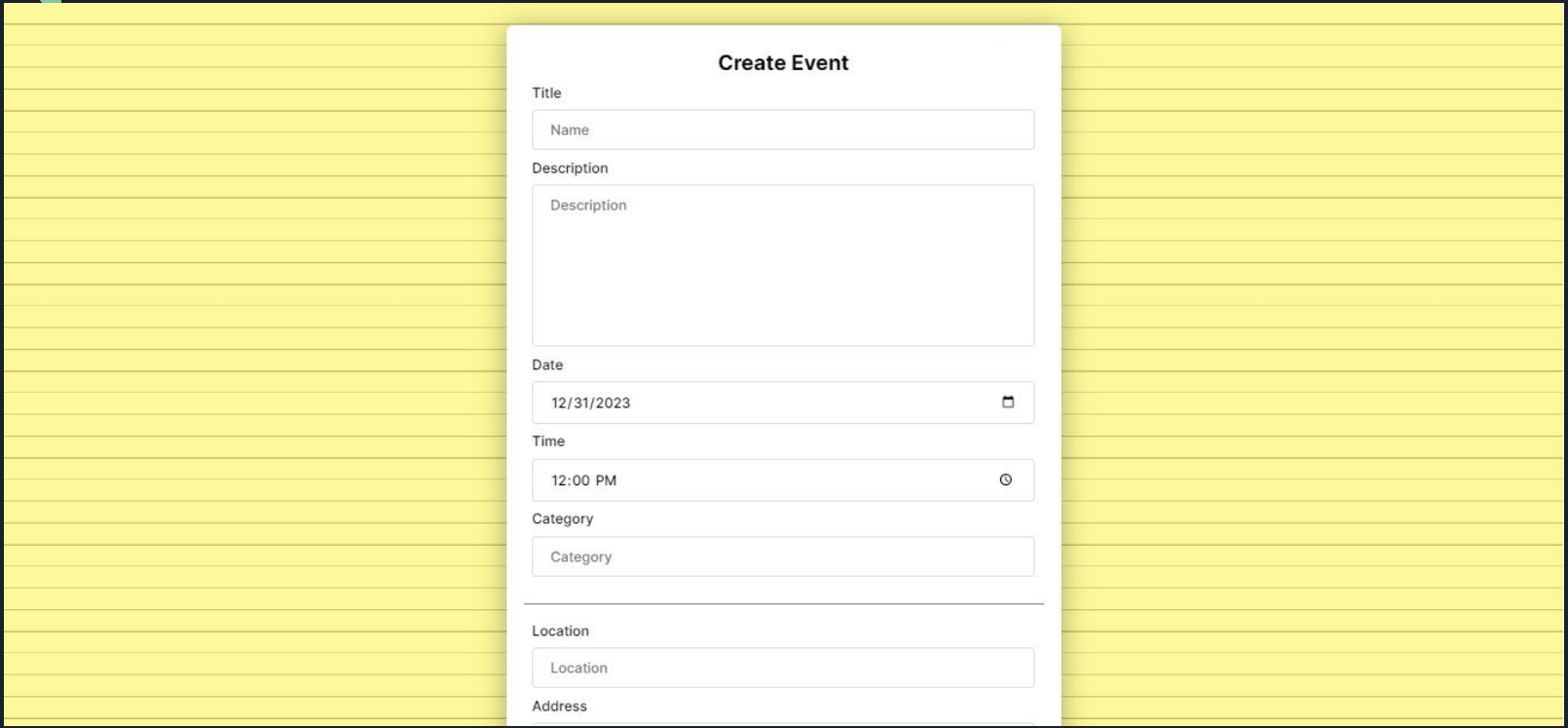
The DBMS I used was phpMyAdmin.



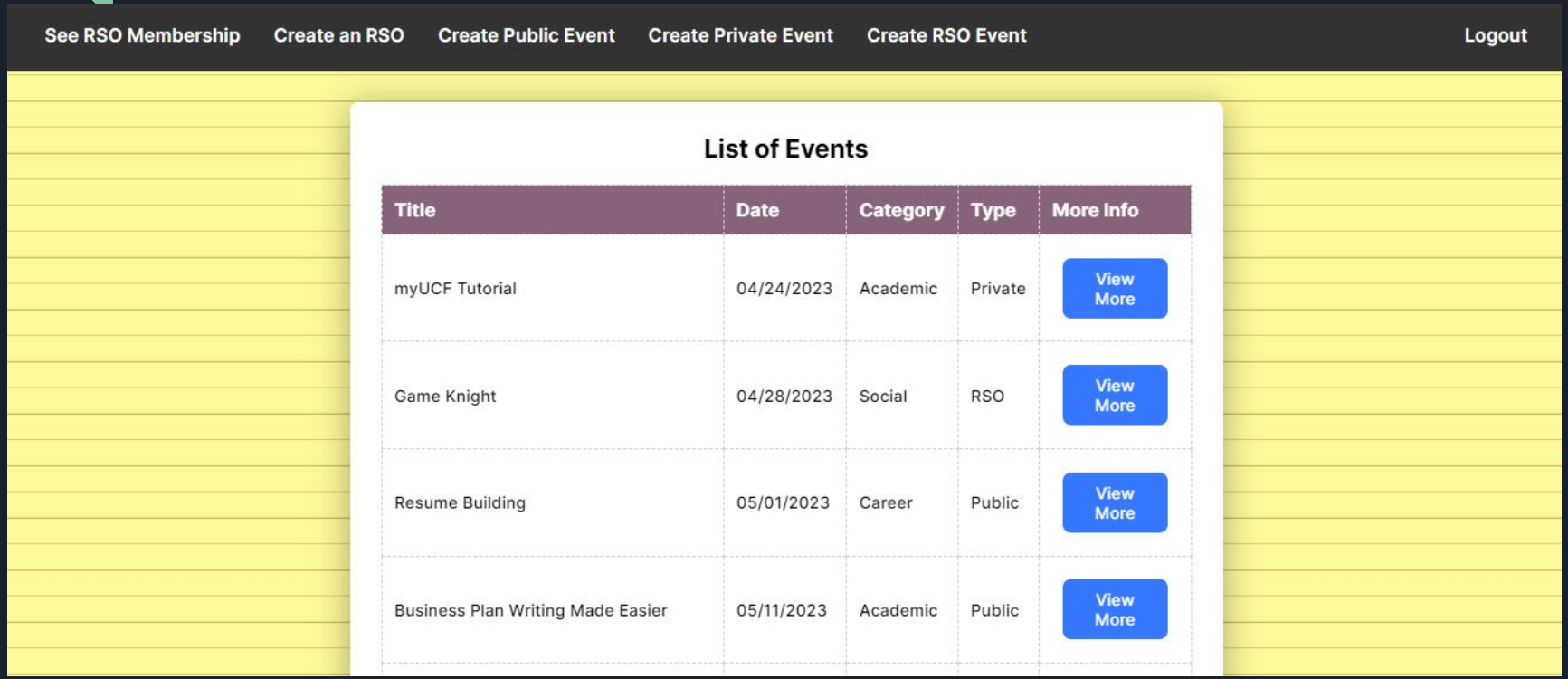
GUI (Create RSO)



GUI (Create Event)

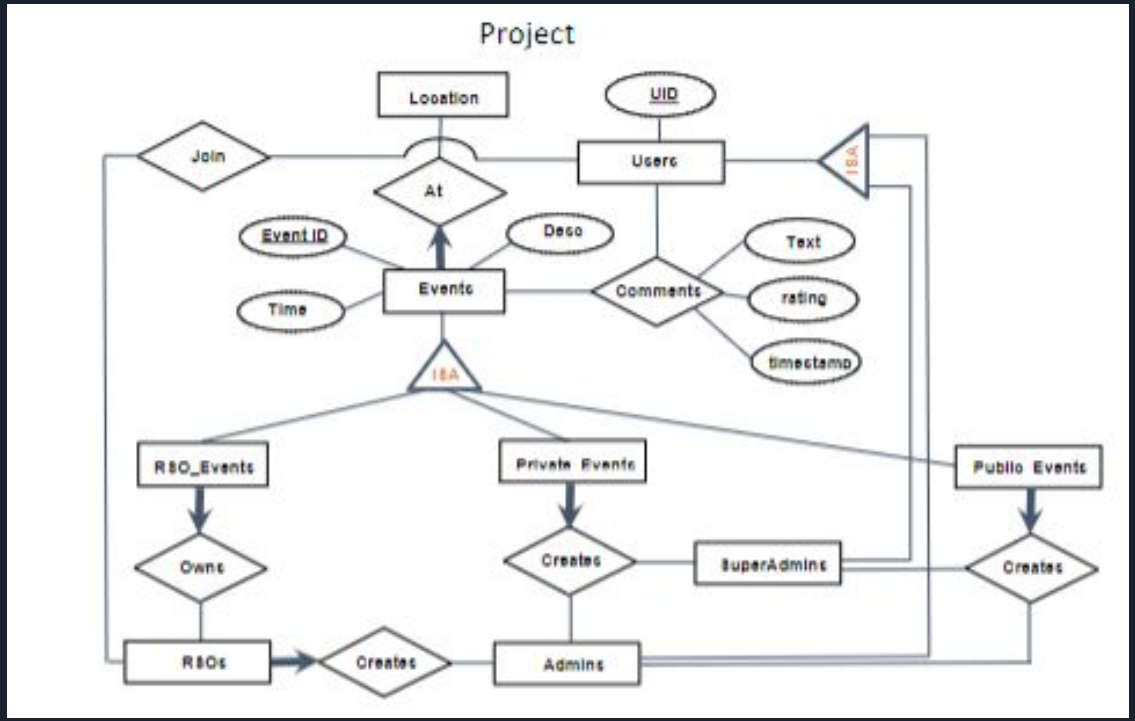


GUI (View Events)

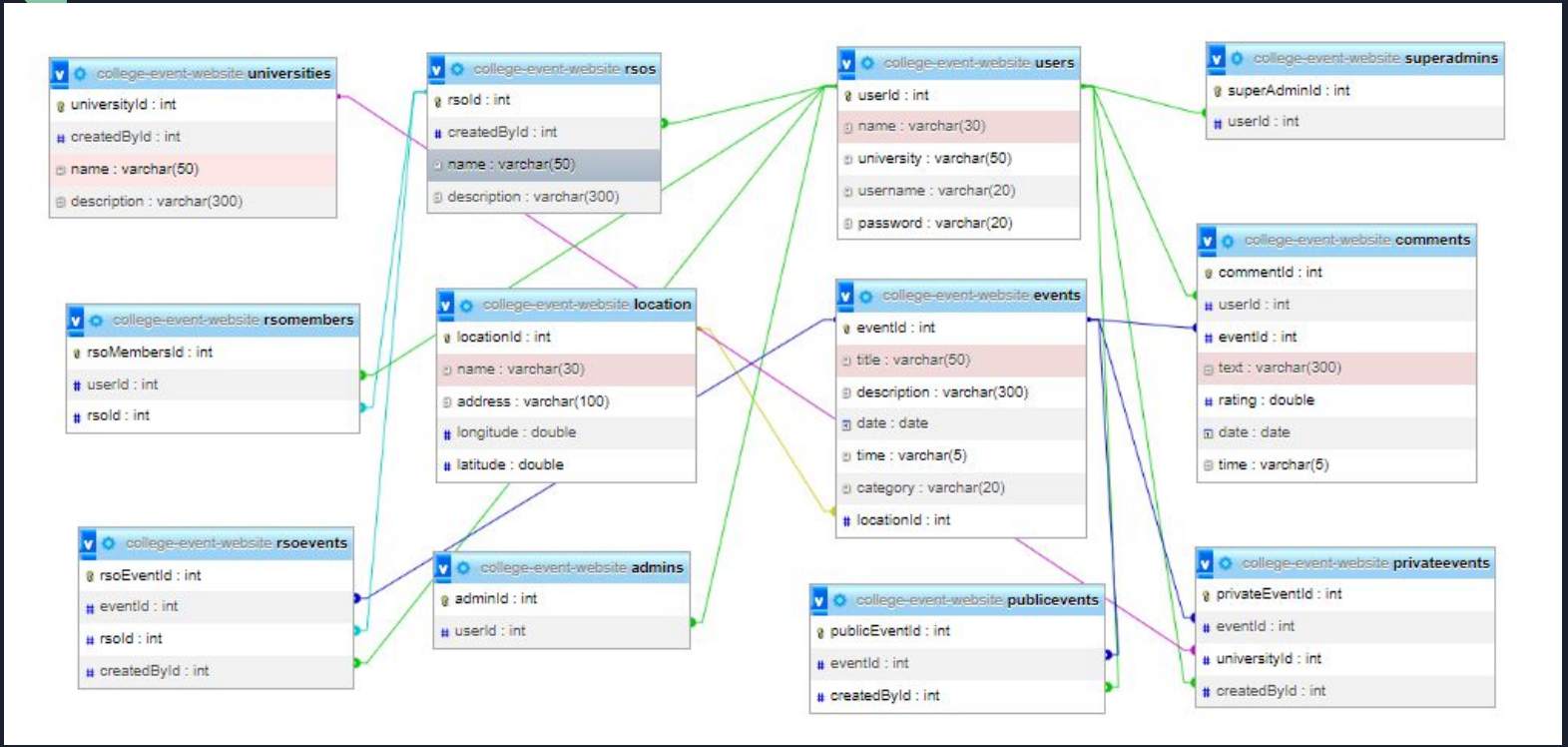


Entity Relationship Diagram

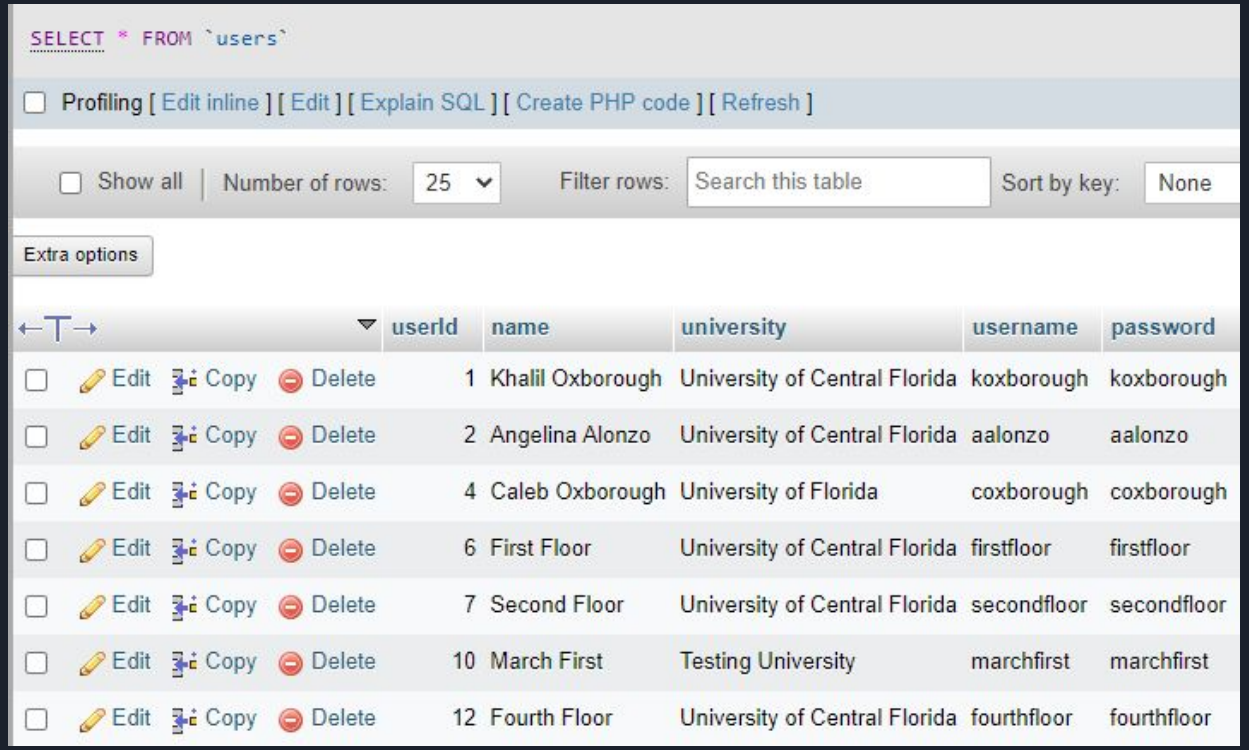
One entity is missing from this diagram, which is the Universities entity. This would be linked to the Users and Events entities through a Foreign Key constraint.



Relational Model



Sample Data (Users)



```
SELECT * FROM `users`
```

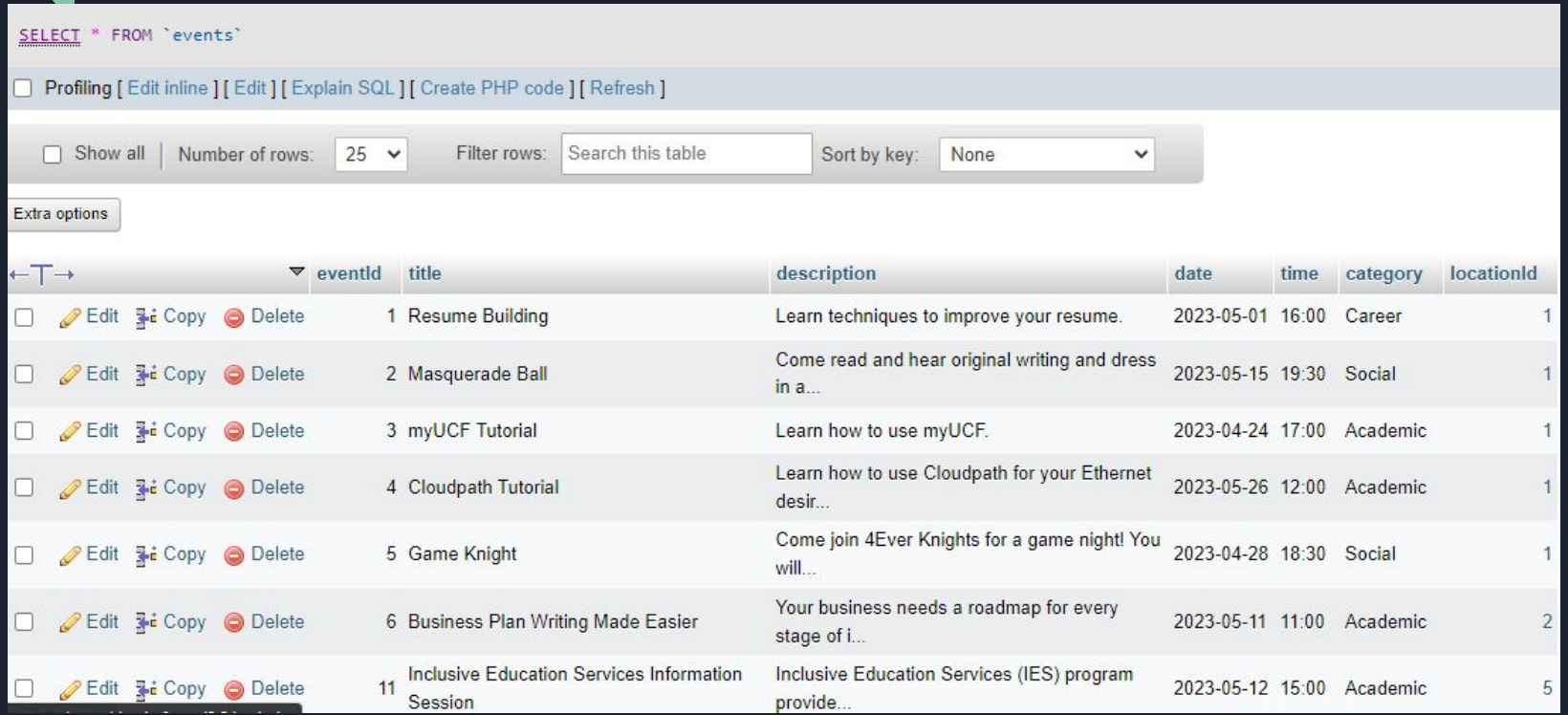
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

	userId	name	university	username	password
Edit Copy Delete	1	Khalil Oxborough	University of Central Florida	koxborough	koxborough
Edit Copy Delete	2	Angelina Alonzo	University of Central Florida	aalonzo	aalonzo
Edit Copy Delete	4	Caleb Oxborough	University of Florida	coxborough	coxborough
Edit Copy Delete	6	First Floor	University of Central Florida	firstfloor	firstfloor
Edit Copy Delete	7	Second Floor	University of Central Florida	secondfloor	secondfloor
Edit Copy Delete	10	March First	Testing University	marchfirst	marchfirst
Edit Copy Delete	12	Fourth Floor	University of Central Florida	fourthfloor	fourthfloor

Sample Data (Events)



```
SELECT * FROM `events`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

	eventId	title	description	date	time	category	locationId
Edit Copy Delete	1	Resume Building	Learn techniques to improve your resume.	2023-05-01	16:00	Career	1
Edit Copy Delete	2	Masquerade Ball	Come read and hear original writing and dress in a...	2023-05-15	19:30	Social	1
Edit Copy Delete	3	myUCF Tutorial	Learn how to use myUCF.	2023-04-24	17:00	Academic	1
Edit Copy Delete	4	Cloudpath Tutorial	Learn how to use Cloudpath for your Ethernet desir...	2023-05-26	12:00	Academic	1
Edit Copy Delete	5	Game Knight	Come join 4Ever Knights for a game night! You will...	2023-04-28	18:30	Social	1
Edit Copy Delete	6	Business Plan Writing Made Easier	Your business needs a roadmap for every stage of i...	2023-05-11	11:00	Academic	2
Edit Copy Delete	11	Inclusive Education Services Information Session	Inclusive Education Services (IES) program provide...	2023-05-12	15:00	Academic	5

Sample Data (Comments)

```
SELECT * FROM `comments`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

	commentId	userId	eventId	text	rating	date	time
Edit Copy Delete	1	1	1	Looking forward to it! My resume needs some seriou...	5	2023-03-02	14:00
Edit Copy Delete	2	1	3	My skills with myUCF have been slipping, I need he...	4.5	2023-03-03	15:00
Edit Copy Delete	3	1	5	I hope we play Bingo!	4.75	2023-03-06	16:00
Edit Copy Delete	4	1	5	Can't wait to meet the ambassadors!	4	2023-03-06	19:00
Edit Copy Delete	5	1	1	My resume is all good, unfortunately for you guys.	2	2023-03-09	22:02

SQL Examples

1 row inserted.
Inserted row id: 4 (Query took 0.0026 seconds.)

```
INSERT INTO RSOs (createdById, name, description) VALUES (1, 'Test', 'Test');
```

[Edit inline] [Edit] [Create PHP code]

1 row inserted.
Inserted row id: 12 (Query took 0.0024 seconds.)

```
INSERT INTO RSOMembers (userId, rsoId) VALUES (1, 2);
```

[Edit inline] [Edit] [Create PHP code]

1 row inserted.
Inserted row id: 14 (Query took 0.0083 seconds.)

```
INSERT INTO Events (title, description, date, time, category, locationId) VALUES ('Test', 'Test', '01-01-1970', '12:00', 'Test', 1);
```

[Edit inline] [Edit] [Create PHP code]

SQL Examples (cont.)

1 row inserted.
Inserted row id: 6 (Query took 0.0040 seconds.)

```
INSERT INTO Comments (userId, eventId, text, rating, date, time) VALUES (1, 1, 'Test', 2.5, '01-01-1970', '12:00');
```

[Edit inline] [Edit] [Create PHP code]

Showing rows 0 - 5 (6 total, Query took 0.0024 seconds.)

```
CALL GetAllEvents(1);
```

[Edit inline] [Edit] [Create PHP code]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

eventId	title	date	time	category	type
3	myUCF Tutorial	2023-04-24	17:00	Academic	Private
5	Game Knight	2023-04-28	18:30	Social	RSO
1	Resume Building	2023-05-01	16:00	Career	Public
6	Business Plan Writing Made Easier	2023-05-11	11:00	Academic	Public
11	Inclusive Education Services Information Session	2023-05-12	15:00	Academic	Public
2	Masquerade Ball	2023-05-15	19:30	Social	Public

SQL Examples (Stored Procedure)

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `GetAllEvents`(IN `givenId` INT) NOT DETERMINISTIC CONTAINS SQL SQL SECURITY DEFINER BEGIN
(SELECT E.eventId, E.title, E.date, E.time, E.category, 'Public' AS type FROM Events E, PublicEvents P WHERE E.eventId=P.eventId) UNION
(SELECT E.eventId, E.title, E.date, E.time, E.category, 'Private' AS type FROM Universities Un, Users Us, PrivateEvents P, Events E WHERE
Us.userId=givenId AND Us.university=Un.name AND Un.universityId=P.universityId AND P.eventId=E.eventId) UNION (SELECT E.eventId, E.title,
E.date, E.time, E.category, 'RSO' AS type FROM Users U, RSOMembers RSOM, RSOEvents RSOE, Events E WHERE U.userId=givenId AND
U.userId=RSOM.userId AND RSOM.rsoId=RSOE.rsoId AND RSOE.eventId=E.eventId) ORDER BY date, time; END
```

Because I was a one-person team, there were no advanced features added to the project. However, because of a one-person team, no advanced features were required.

Conclusion/Observations

- Query response time and overall performance issues were never a problem, as there are no points in the project where the frontend is lagging as it is waiting for data.
- One feature I desired that I did no implement was security. The passwords are stored in plaintext, and had I had more time I would've implemented this feature.
    - With that being said, I am proud of the project and I have no other desired features than above.
- If I were to go back in time, I would have put more time into planning the functionality and organization much better. I more or less worked at a continuous pace and implemented features as I went along, and sometimes I would have to refactor and fix features that I already wrote.