

# University Events Website

COP 4710 - 0001, UCF Spring 2017

Group 47

04.17.17

**Author:** Guillermo Alicea

## Table of Contents

- I. Project Description**
  - A. Assumptions
- II. User Interface**
  - A. Languages/DBMS Used
  - B. Screenshots of Application and DBMS
- III. ER-Model**
  - A. Enforcement of Constraints
- IV. Database Schema**
- V. Sample Data Population**
  - A. Screenshots of Sample Data in Database
- VI. SQL Examples**
- VII. Software Installation “How-to”**
  - A. Setting up Environment/Server
  - B. Setting up Database
- VIII. Observations**
  - A. Desired Features
  - B. Problems encountered
  - C. Database Performance
- IX. Conclusion**

## Project Description

The course project for COP4710, Spring 2017, is one that, to my knowledge, has been given in the past, in similar but not identical fashion. It is a relatively simple application, laced with constraints and mandatory features that must be satisfied as a set of requirements for the system. The underlying aim of the project is to demonstrate knowledge of, and the ability to implement, a web application that connects to and stores/receives information from a database. More particularly, the project aims to allow students to demonstrate their ability to design and create a well-made schema that must satisfy a given set of requirements, while also creating queries that will correctly update and retrieve data from the database.

The official project description is as follows:

You are asked to implement a web application that solves the aforementioned problems. Any student may register with this 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. In order 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 student user can request to create a new RSO or to join an existent one. A new RSO can be created with at least 5 other students with the same email domain, e.g. @knights.ucf.edu; and one of them should be assigned as an administrator.

Student can view events in their university by location, or by selecting the University they want to see the events from. They can retrieve events according to their level of access or scope. A student should be able to see all the events around its location or from RSOs they are following.

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 of 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. Such events must be approved by the super admin. After an event has been published, users can add, remove, and edit comments on the event, as well as rating the event with up to 5 stars. The application should offer some social network integration, e.g. posting from the application to Facebook or Google.

Additionally, a number of technical requirements are put in place as part of the assignment. They include:

1. The software must include at least 5 relational tables.
2. The software must include at least 10 SQL queries.
3. The website and database must support multiple concurrent users.
4. The application must have a browser-based interface and can be deployed on Internet.
5. The capabilities mentioned in the project description are worth 80 points (out of 100 points). Each team can design and implement additional features for the remaining 20 points. Extra credits will be given to outstanding user interface and special features.
6. Programming languages that can be used for the project: HTML, Javascript, PHP, Java, CSS, and stored procedures. DBMS's: Oracle, SQL Server, and MySQL. Other languages and DBMSs: check with the instructor.

As you will come to find, both the functional and technical requirements have been met in full throughout the development of this project.

### Assumptions

Before continuing on with the in-depth description of this project, it is necessary for me to discuss the assumptions that were made throughout the development process. The following assumptions are not overwhelming in impact or use of the application, and they do not cause the violation of any of the aforementioned requirements - they simply serve the purpose of emphasizing the main purpose of the project, and overshadowing the aspects of the project which are clearly not meant to be in the forefront. They include:

1. The user is using a browser which is listed in the top 5 most popularly used, and it is up to date. Throughout testing, Google Chrome has been used.
2. Certain features of the project require Javascript, thus it is assumed that the user will have Javascript enabled on their browser.

3. It is assumed that users of the system have traits of integrity and honesty, and thus will use the system with those in mind. Namely, they will sign up for the system with correct user types given their relative standing, and they will provide valid - and correct - input wherever it is required/asked for.

4. The system has been developed on Linux, so it is assumed that the user will also use this OS. This assumption may not matter, however, as it is unlikely that the application will cease to work if a different OS is used.

5. The user will be viewing the site on a desktop, though this assumption also may not matter, as it is unlikely that viewing it on a mobile device or laptop will cease its function. The layout of the pages may vary however, though for the most part the application is responsive to such variance.

6. The application is not currently being hosted, so until it is (if it ever will be), it is assumed that the user will be viewing the site on the local machine on which it was developed.

## **User Interface**

### Technical Specifications

The technical decisions made with regards to implementation of the user interface (front-end), logic and functionality (back-end), and data storage and access (database), were made with the following considerations in mind:

1. The tool/language used should already be known, or should be simple to learn to a degree that it can be effectively utilized in the scope of this project.
2. The tool/language should be capable of fulfilling the task(s) for which it will be used.
3. The tool/language must be listed in those that are allowed to be used for this project (in assignment description)

With this in mind, I have decided to implement the project using the following tools/languages:

- Front-end: HTML5, CSS3, Javascript (some AJAX, facebook integration, and google maps integration), Bootstrap (for buttons, navigation menu)

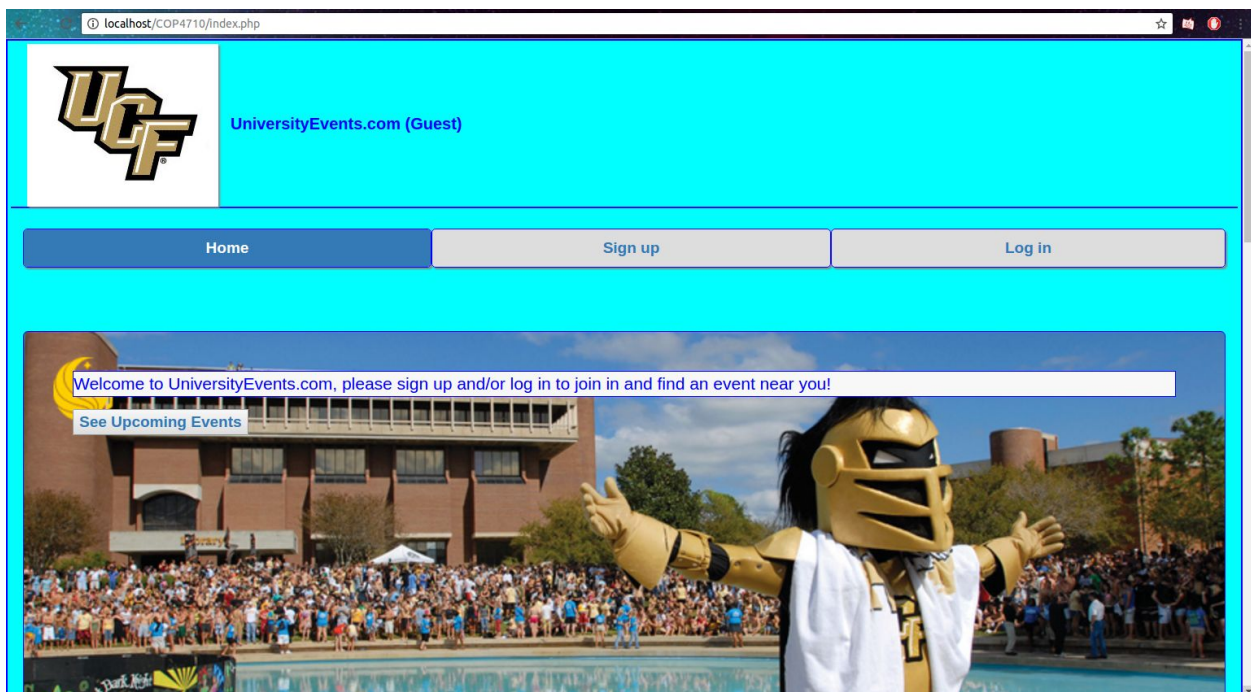
- Back-end: LAMP stack, PHP 5.3 (used for essentially all logic and application-server side functionality)
- Database: MySQL Server, manipulation of Database was done on the command-line

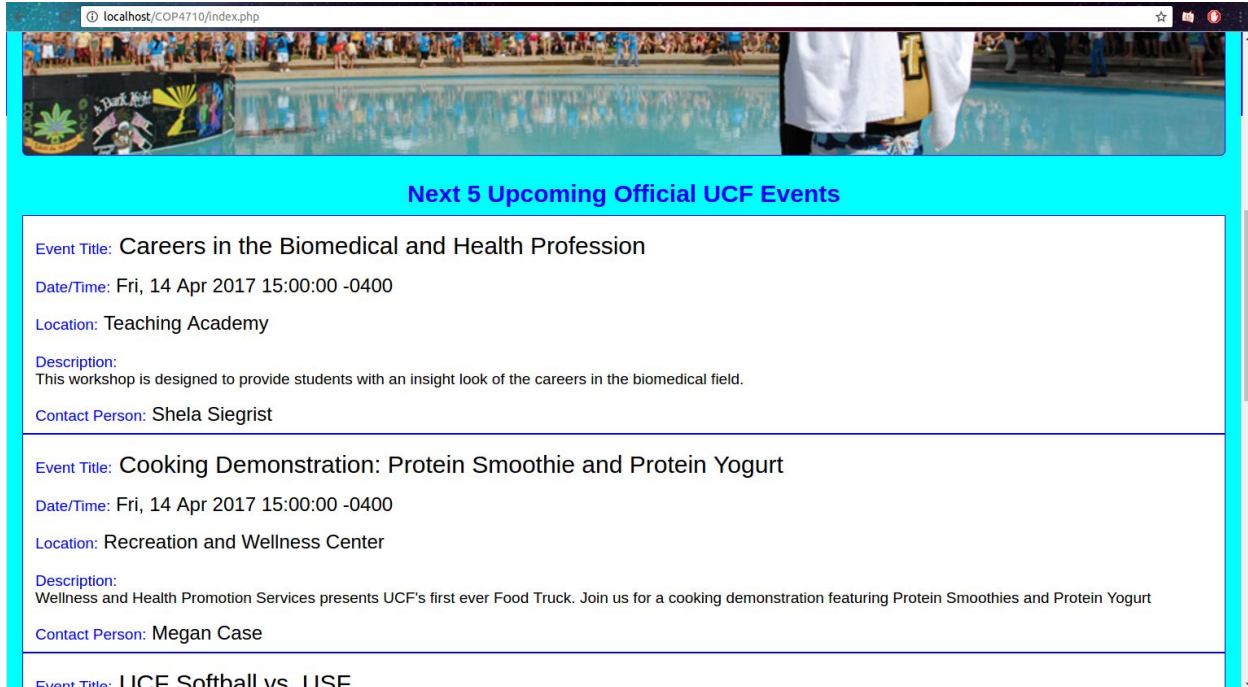
### UI Screenshots

Given my lack of experience with web design (and design in general), the GUI of this site is not very modern/sleek, especially in comparison to professionally designed sites. However, this was not a major part of the project anyhow, and the functionality of the system was achieved to its entirety, so the lack of a quality UI should not be a major issue.

#### *Home Screen (not logged in)*

The home screen, when the user is not logged in, displays the same for everyone. It is a simple page with a navigation menu, a list of 5 upcoming events, and the footer.





**Next 5 Upcoming Official UCF Events**

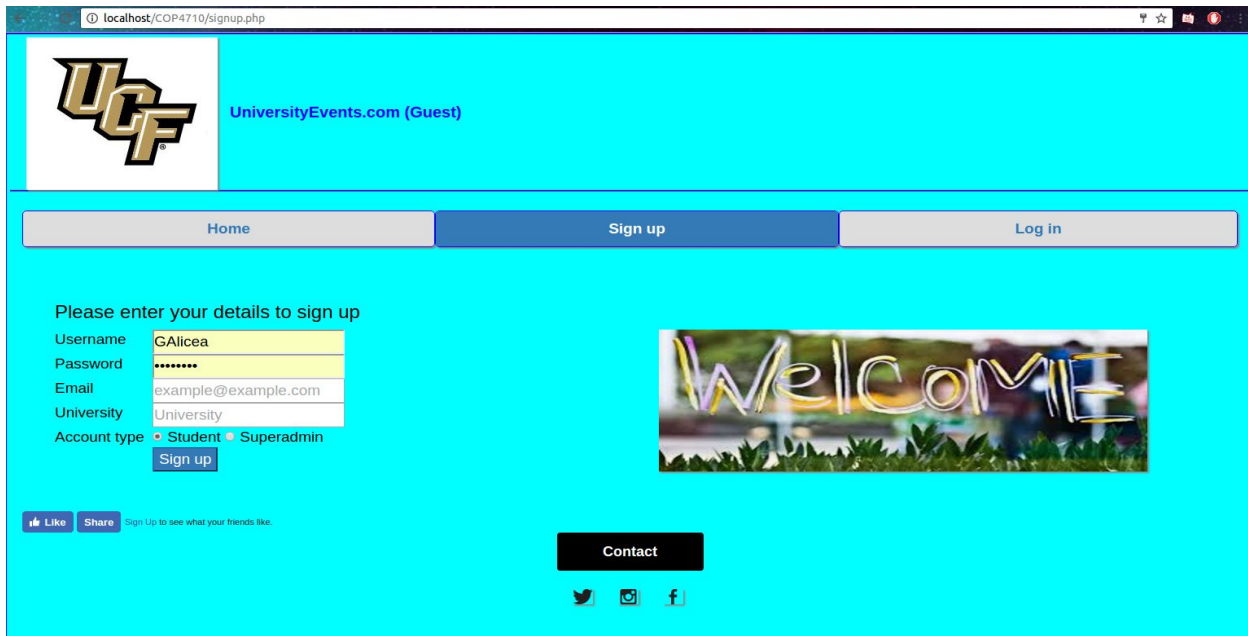
**Event Title:** Careers in the Biomedical and Health Profession  
**Date/Time:** Fri, 14 Apr 2017 15:00:00 -0400  
**Location:** Teaching Academy  
**Description:** This workshop is designed to provide students with an insight look of the careers in the biomedical field.  
**Contact Person:** Shela Siegrist

**Event Title:** Cooking Demonstration: Protein Smoothie and Protein Yogurt  
**Date/Time:** Fri, 14 Apr 2017 15:00:00 -0400  
**Location:** Recreation and Wellness Center  
**Description:** Wellness and Health Promotion Services presents UCF's first ever Food Truck. Join us for a cooking demonstration featuring Protein Smoothies and Protein Yogurt  
**Contact Person:** Megan Case

**Event Title:** UCF Softball vs. USF

### Signup

The signup page is a regular page where the user can input general account information, except it will also ask him to input his account type - either student or superadmin (admins are made when students make RSOs). If the type is selected as superadmin - the university that is used in account creation will be made created and an associated profile for the university will be made, unless it is already created, in which case the superadmin will not be able to use that university. If the type is student, then a superadmin must have already created the student's university, or else the student cannot make an account with that university.



**UCF** UniversityEvents.com (Guest)

Home Sign up Log in

Please enter your details to sign up

Username: GAlieca  
 Password: \*\*\*\*\*  
 Email: example@example.com  
 University: University  
 Account type: ☒ Student ☐ Superadmin  
 Sign up

Welcome

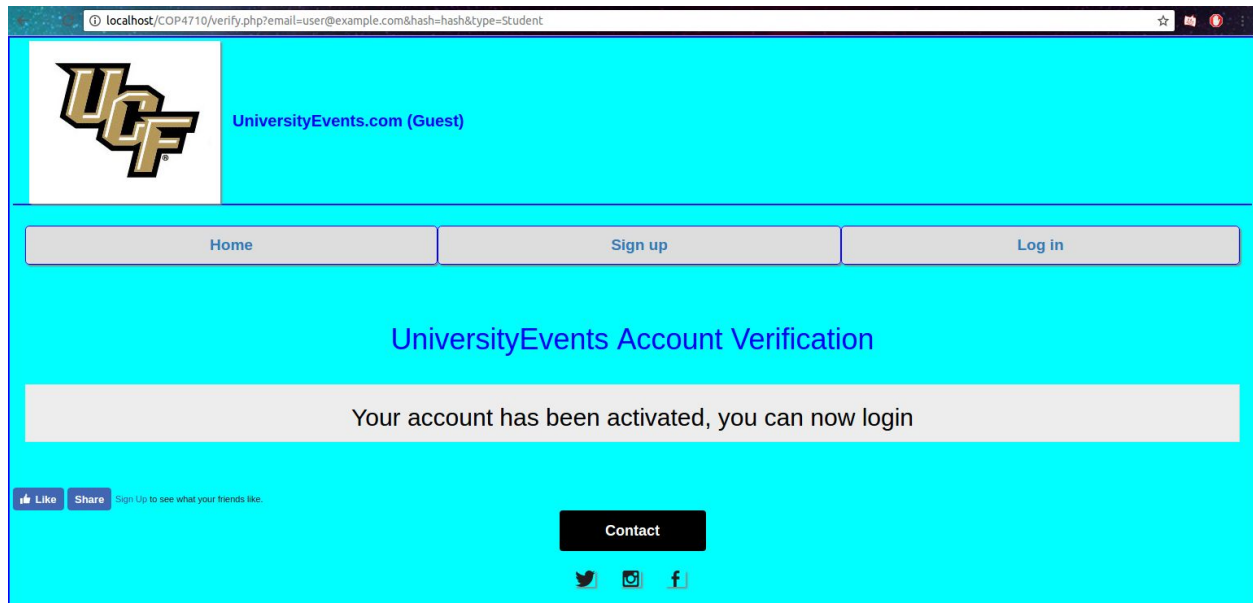
Contact

Like Share Sign Up to see what your friends like.

Twitter Instagram Facebook

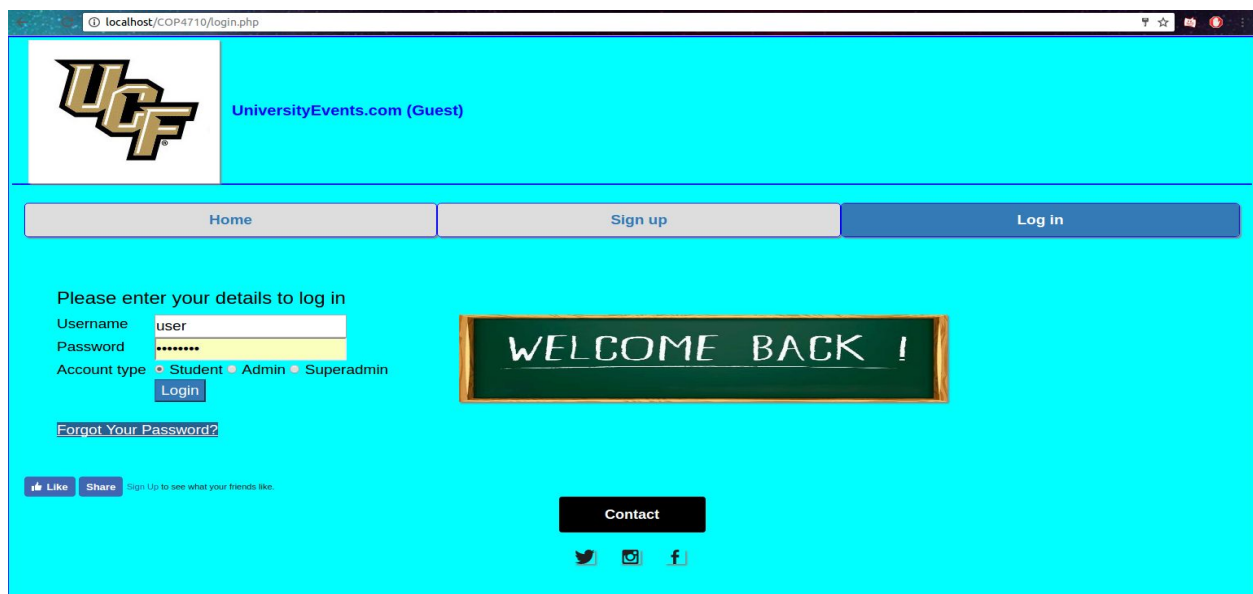
## Verify

After creating an account, an email will be sent to the email used for registration [if the site were hosted on a web server - since it is not, the code instead is set to simply use 'hash' as what would normally be a randomly hashed value]. A link can then be followed (input manually now since an email cannot be sent), which will lead the user to this page - if the url is valid, the account will then be activated, otherwise an error is given to the user.



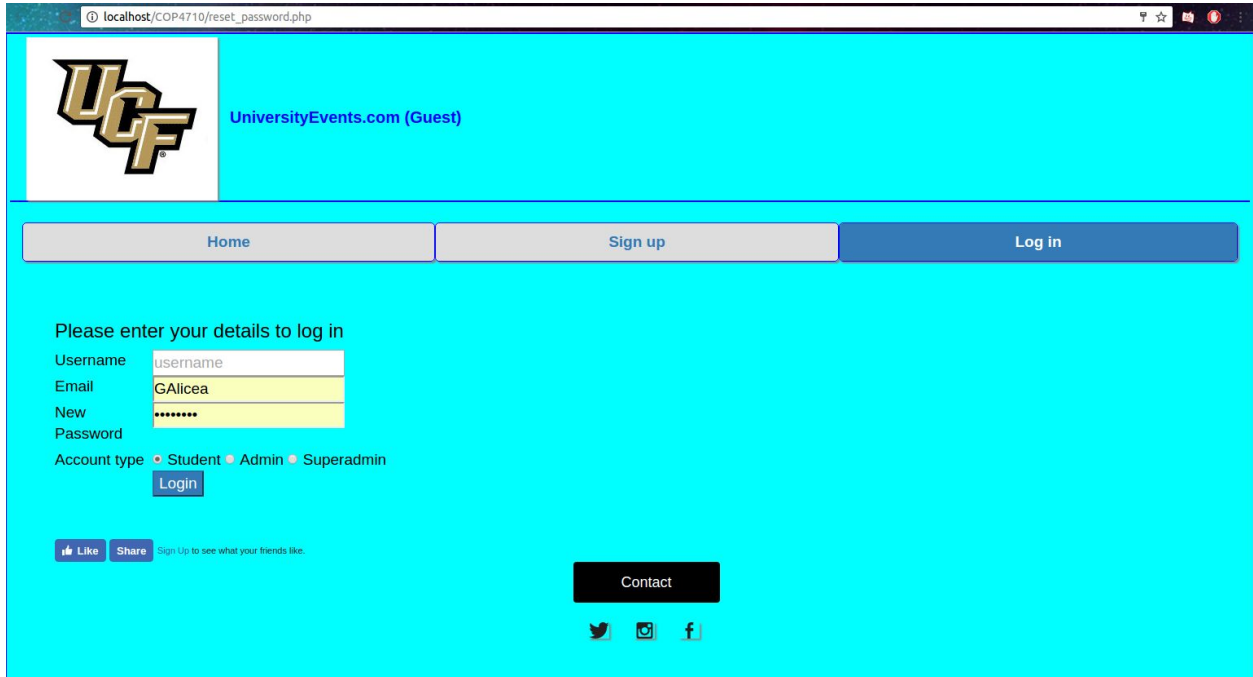
## Login

This is a simple login page that the user can use to login - they will need to specify which account type they have set for their account, as they can only login as that type.



## Password Reset

This is just a simple page where the user - if they input the correct username and email combination - can reset their password.



localhost/COP4710/reset\_password.php

UniversityEvents.com (Guest)

Home Sign up Log in

Please enter your details to log in

Username

Email




New Password

Account type ☒ Student ☐ Admin ☐ Superadmin

Login

Like Share Sign Up to see what your friends like.

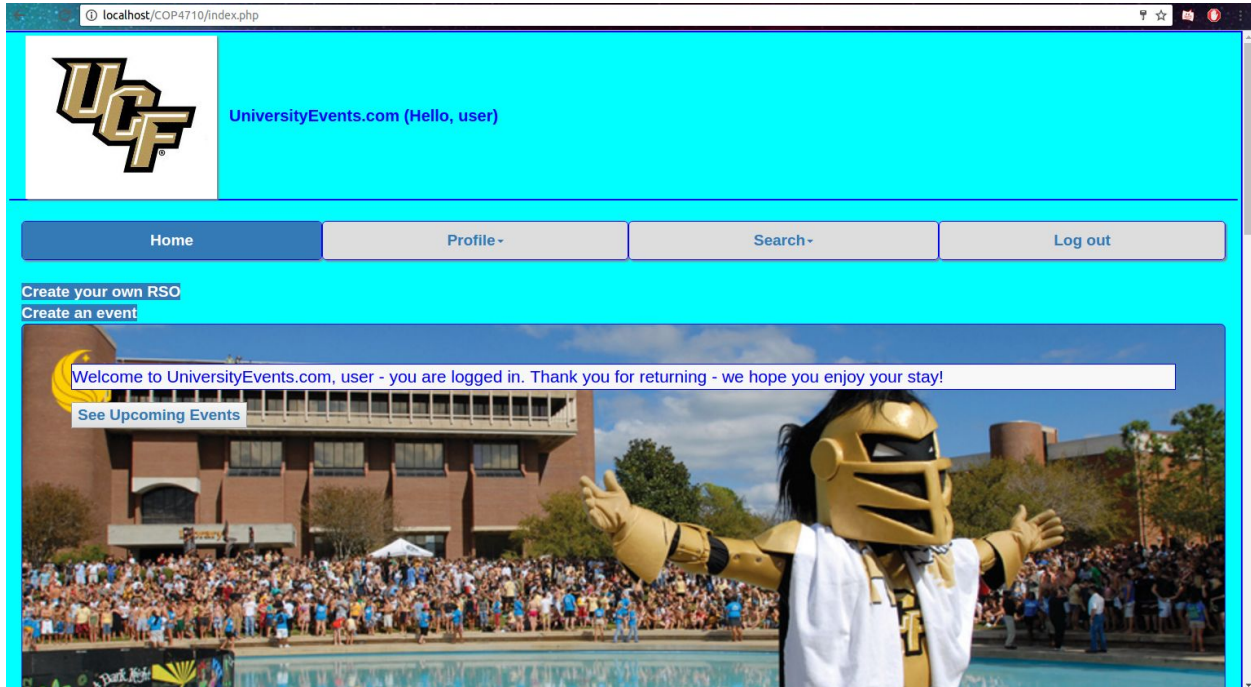
Contact



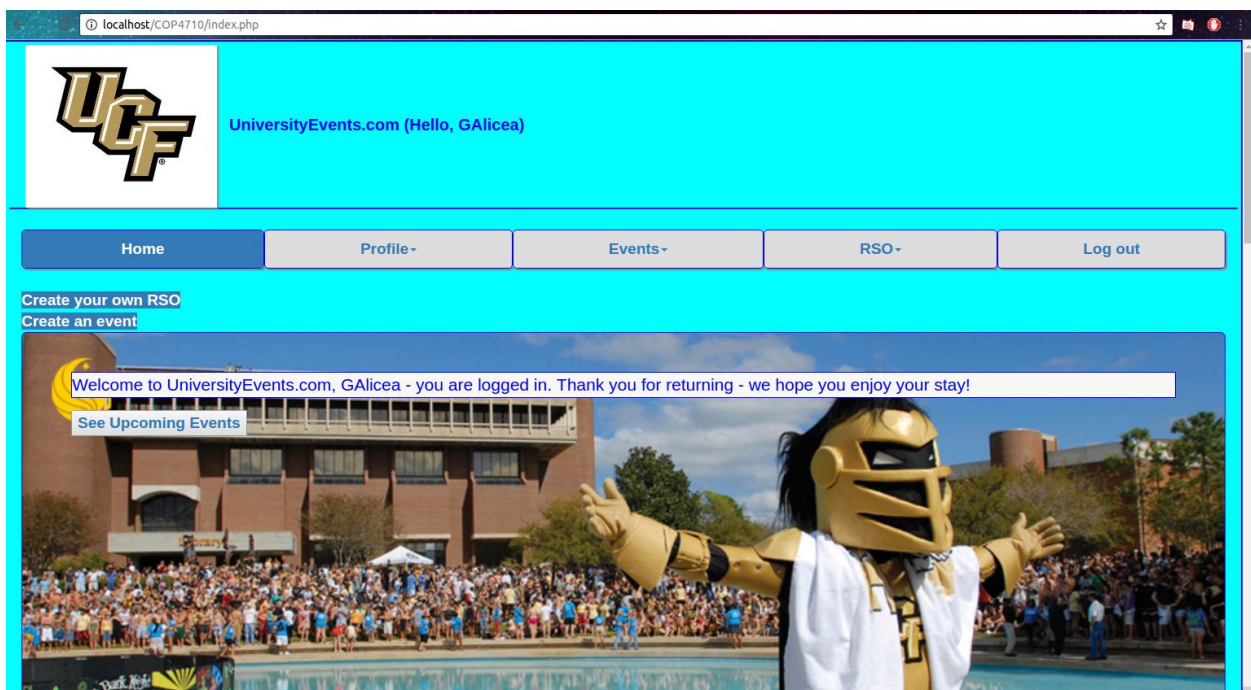
### Home Screen (logged in - Student)

This is the home screen displayed for a user that is logged in as a student. The navigation menu will be the same for every other page they visit.



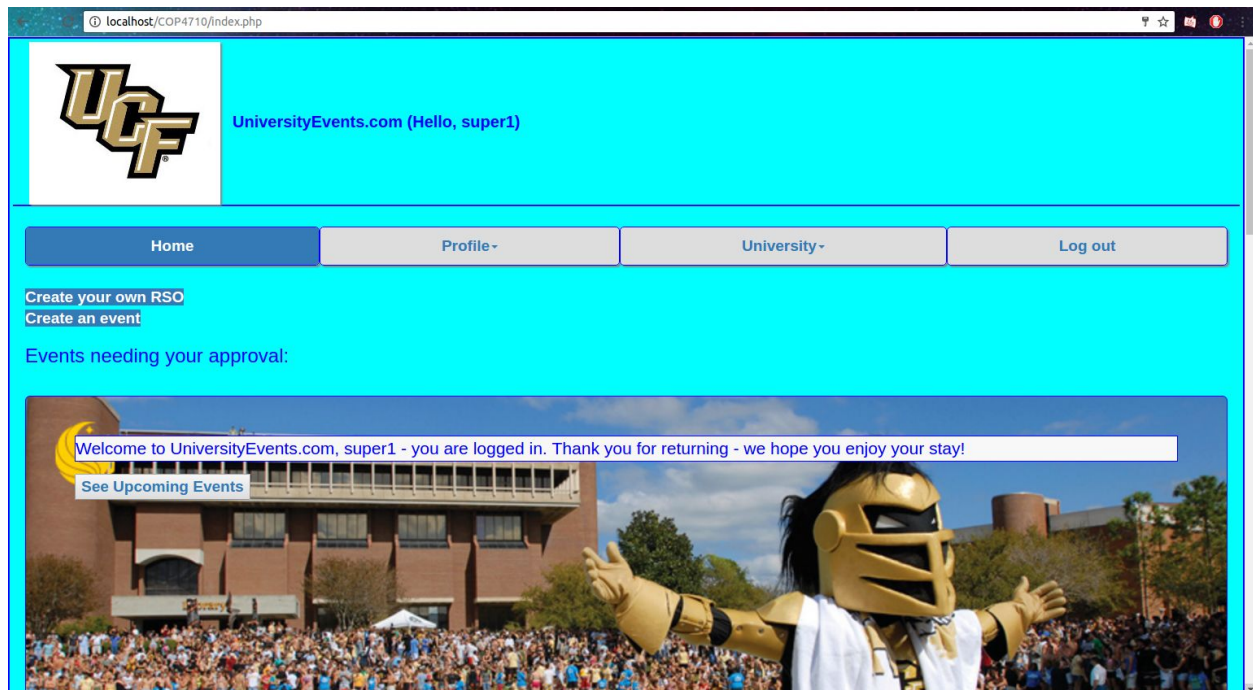
### Home Screen (logged in - Admin)

This is the home screen displayed for a user that is logged in as an admin. The navigation menu will be the same for every other page they visit.



### Home Screen (logged in - Superadmin)

This is the home screen displayed for a user that is logged in as a superadmin. The navigation menu will be the same for every other page they visit. Additionally, per the requirements of the project, any events needing the approval of the superadmin will be listed on their home page.




### View Profile (User)

This is the profile page that can be viewed by any user (all account types have one). Currently, there is no search feature to look for users; however, members of RSOs are listed on the RSO page, and their name can be clicked - doing so will lead the current user to that member's profile page.

localhost/COP4710/user\_profile.php?user=user

Home Profile - Search - Log out

### user's Profile



RSOs  
Events

About me:  
I am a student

University:  
university of central florida

Age:  
20

Male

Grade:  
Senior

Grad Year:  
2017

Major:  
computer science

Minor (if any):  
math

Followed RSOs

Followed Events

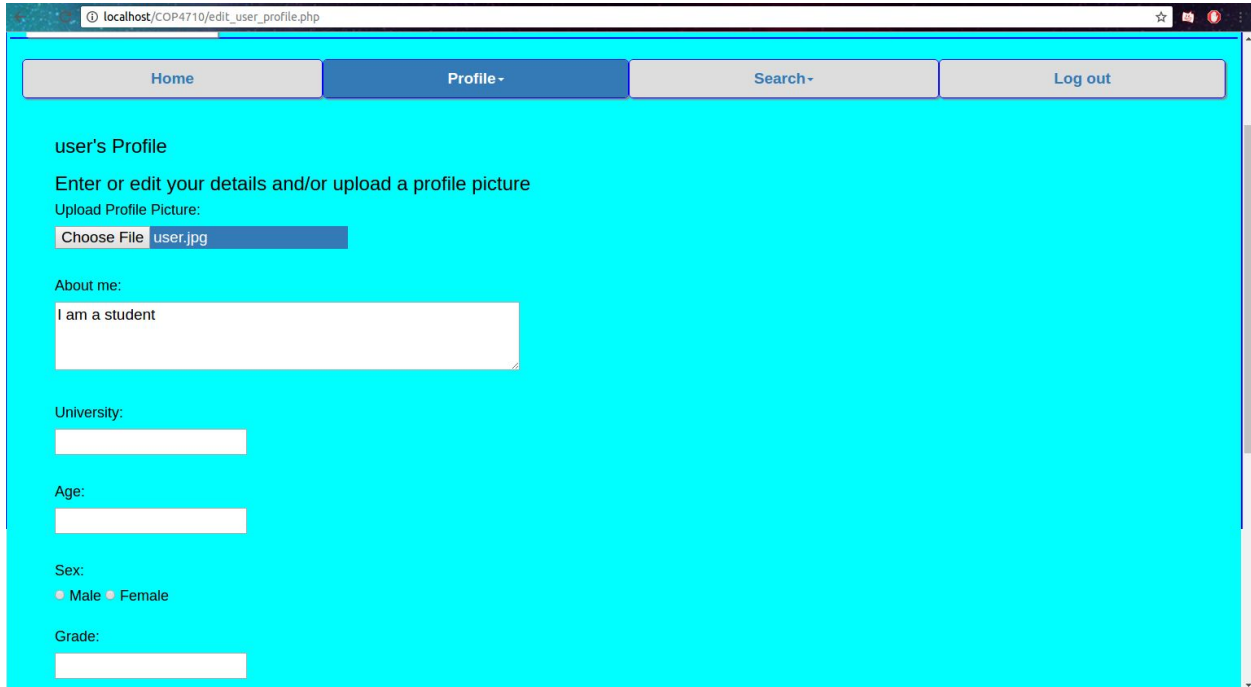
Like Share Sign Up to see what your friends like.

Contact

Twitter Instagram Facebook

## Edit Profile (User)

This is a simple page that will allow the current user to edit their own profile.



localhost/COP4710/edit\_user\_profile.php

Home Profile Search Log out

user's Profile

Enter or edit your details and/or upload a profile picture

Upload Profile Picture:

Choose File user.jpg

About me:

I am a student

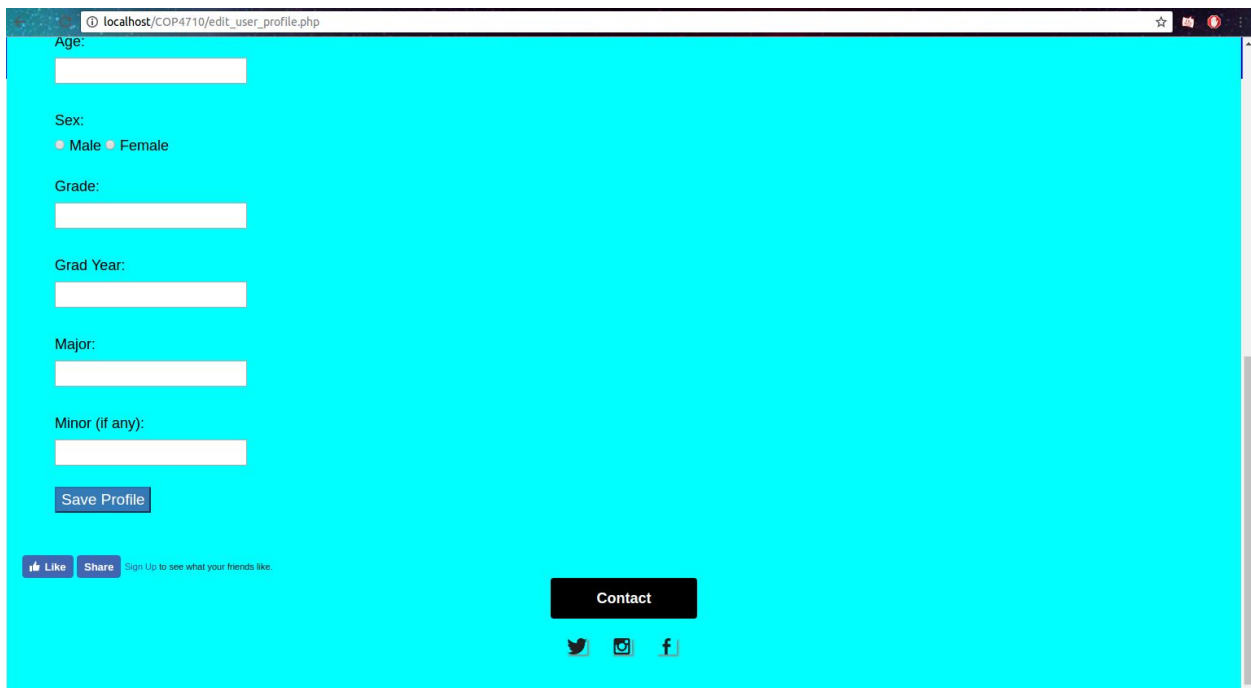
University:

Age:

Sex:

☒ Male ☐ Female

Grade:



localhost/COP4710/edit\_user\_profile.php

Age:

Sex:

☒ Male ☐ Female

Grade:

Grad Year:

Major:

Minor (if any):

Save Profile

Like Share Sign Up to see what your friends like.

Contact

Twitter Instagram Facebook

### *Search RSOs (Student only)*

Students are the only user types which are able to search for RSOs. They can search by two fields: University and Category. They will see general information about the RSO, and can click on a link to go the RSO's page.

The screenshot shows a web browser window with the address bar displaying 'localhost/COP4710/search\_rso.php'. The page has a blue header with a navigation bar containing 'Home', 'Profile ~', 'Search ~', and 'Log out'. Below the navigation bar, there are two input fields: 'University' with the placeholder text 'Do not leave blank' and 'Category' with the placeholder text 'Leave blank for full list'. A 'Search' button is located below these fields. The main content area is titled 'RSOs in University (university of central florida)'. It displays the following information: 'RSO: e', 'Category: e', 'Description:', 'Number of members: 6', and 'Owner: GAlicea'. Below this information is a 'Link to RSO Page' link. At the bottom of the page, there is a 'Like' button, a 'Share' button, and a 'Contact' button. Social media icons for Twitter, Instagram, and Facebook are also present.

localhost/COP4710/search\_rso.php

Home Profile ~ Search ~ Log out

University Do not leave blank  
Category Leave blank for full list  
Search

RSOs in University (university of central florida)

RSO: e  
Category: e  
Description:  
Number of members: 6  
Owner: GAlicea  
[Link to RSO Page](#)


Like Share Sign Up to see what your friends like.

Contact

Twitter Instagram Facebook

## View RSO

This is the page that acts as the 'RSO profile page'. It shows general information about the RSO, as well as giving the user (only students can do this, others will get an error) the option to follow the RSO.



localhost/COP4710/rso\_profile.php?rso=e

Home Profile Search Log out

### e's Profile

Members List  
RSO events

Description:  
e

Owner:  
GAlicea

Type:  
e

Population:  
6

Owner:  
GAlicea

Type:  
e

Population:  
6

### RSO Members

GAlicea

### RSO Events

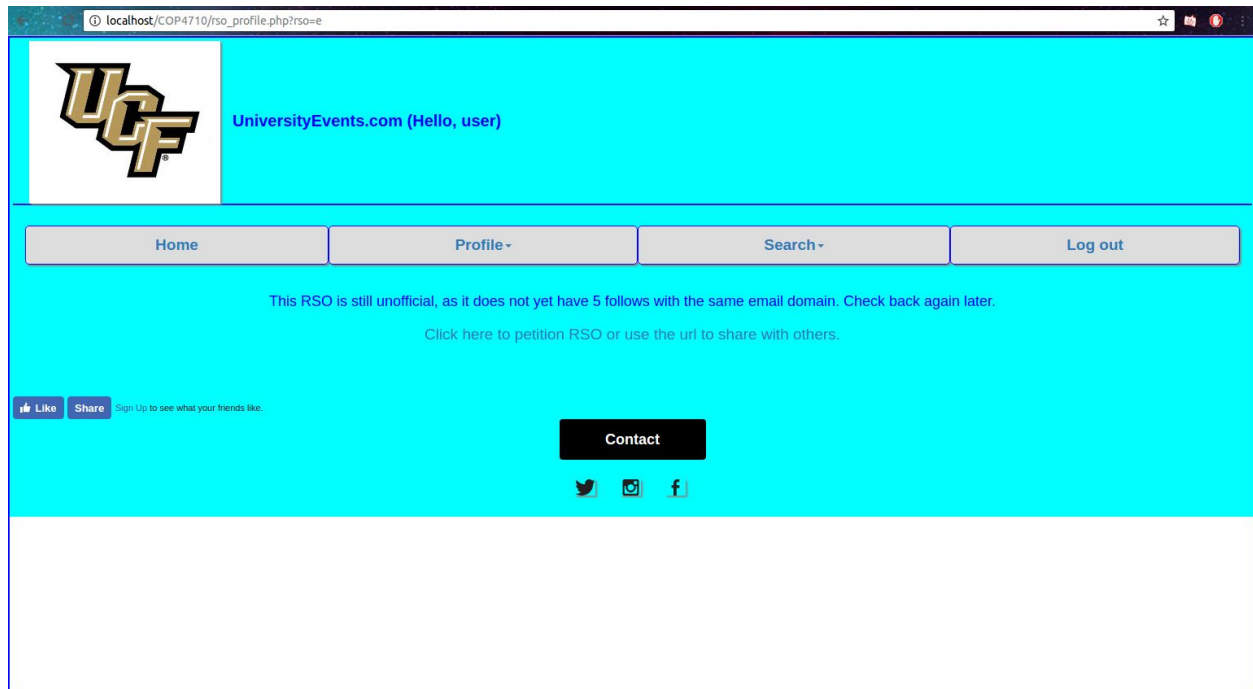
Join RSO

Like Share Sign Up to see what your friends like.

Contact

Twitter Instagram Facebook

If the RSO is inactive (< 5 members), the user will be shown the following page, which provides them with a link to go to the petition page for the RSO.



### *Search Events (Student only)*

Students are the only user types which are able to search for Events. They can search by two fields: University and location. They will see general information about the event, and can click on a link to go to the Event's page. Scope constraints are enforced on this page - the scope of event will determine whether it will be shown on the page, corresponding to the current user's own privileges.

localhost/COP4710/search\_event.php

Home Profile Search Log out

University Do not leave blank  
Location Leave blank for full list  
Search

Events in university of central florida

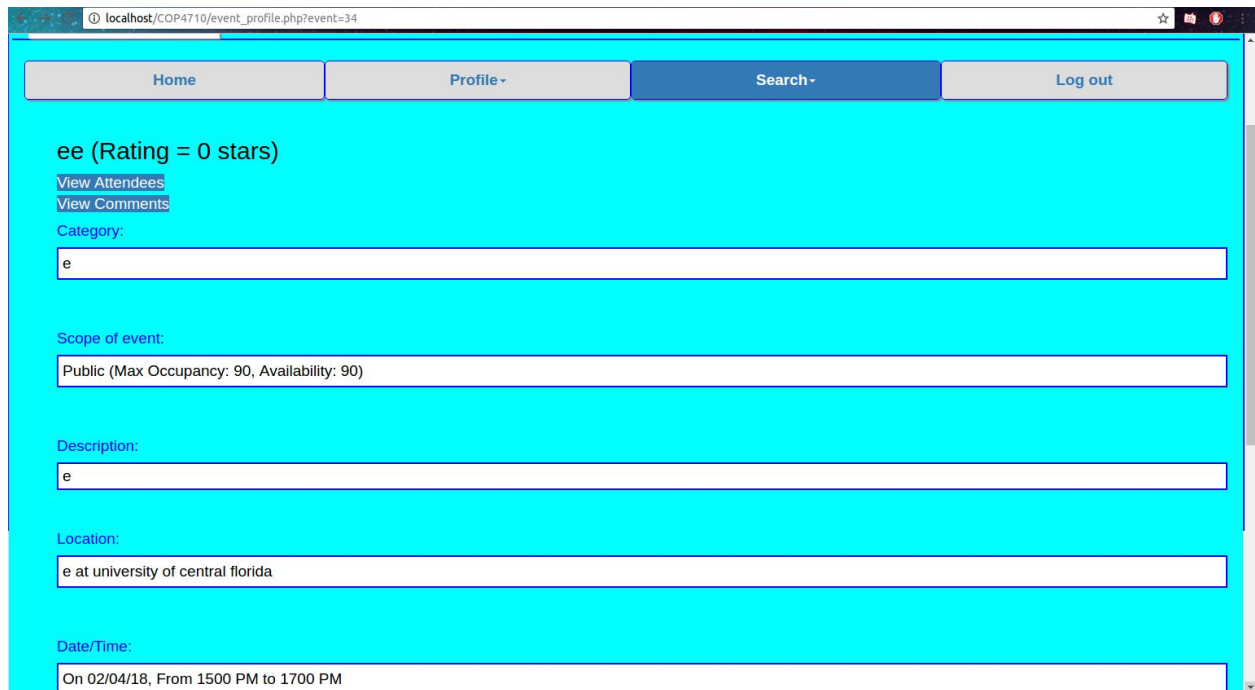
Event Title: edcc (- stars)  
Date/Time: 124312, 1420 AM  
Location: e  
Description: ed  
Contact Person: e  
[Link to Event Page](#)

Event Title: edccs (- stars)  
Date/Time: 124312, 1490 AM  
Location: e  
Description: ed  
Contact Person: e



## View Event

This is the page that acts as the 'Event profile page'. It shows general information about the Event, as well as giving the user (only students can do this, others will get an error) the option to follow the Event.



localhost/COP4710/event\_profile.php?event=34

Home Profile Search Log out

ee (Rating = 0 stars)

[View Attendees](#)

[View Comments](#)

Category:

e

Scope of event:

Public (Max Occupancy: 90, Availability: 90)

Description:

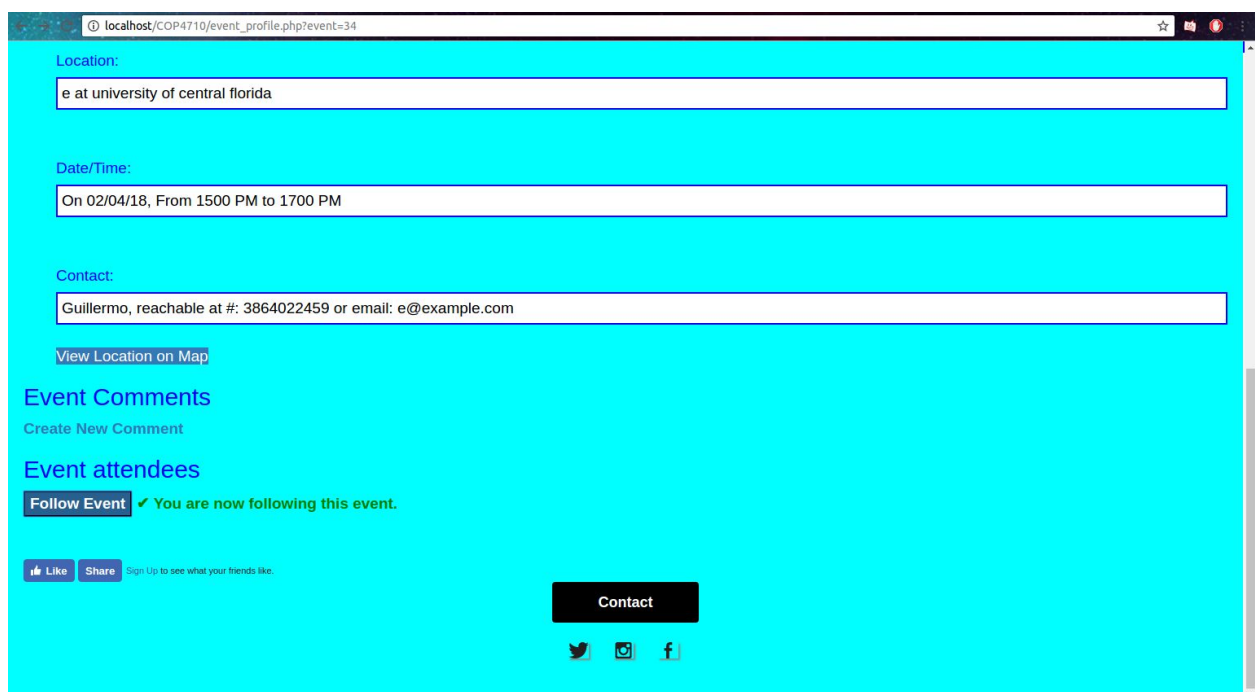
e

Location:

e at university of central florida

Date/Time:

On 02/04/18, From 1500 PM to 1700 PM



localhost/COP4710/event\_profile.php?event=34

Location:

e at university of central florida

Date/Time:

On 02/04/18, From 1500 PM to 1700 PM

Contact:

Guillermo, reachable at #: 3864022459 or email: e@example.com

[View Location on Map](#)

Event Comments

Create New Comment

Event attendees

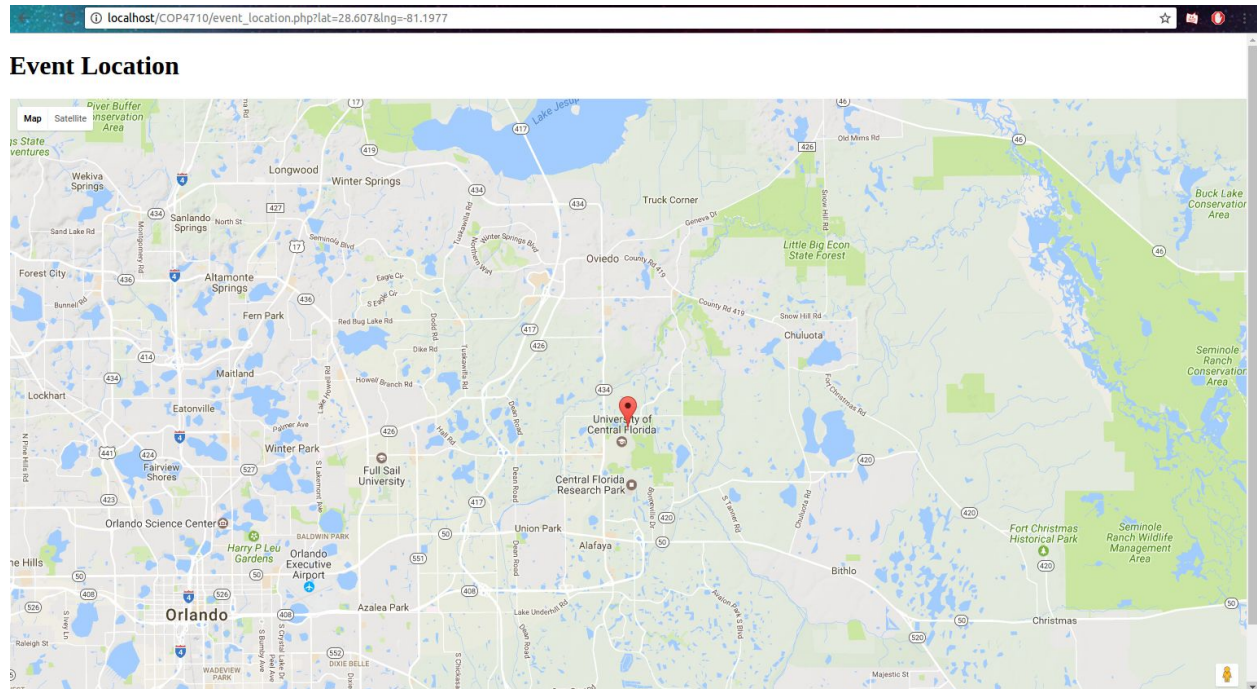
Follow Event ✓ You are now following this event.

Like Share Sign Up to see what your friends like.

Contact

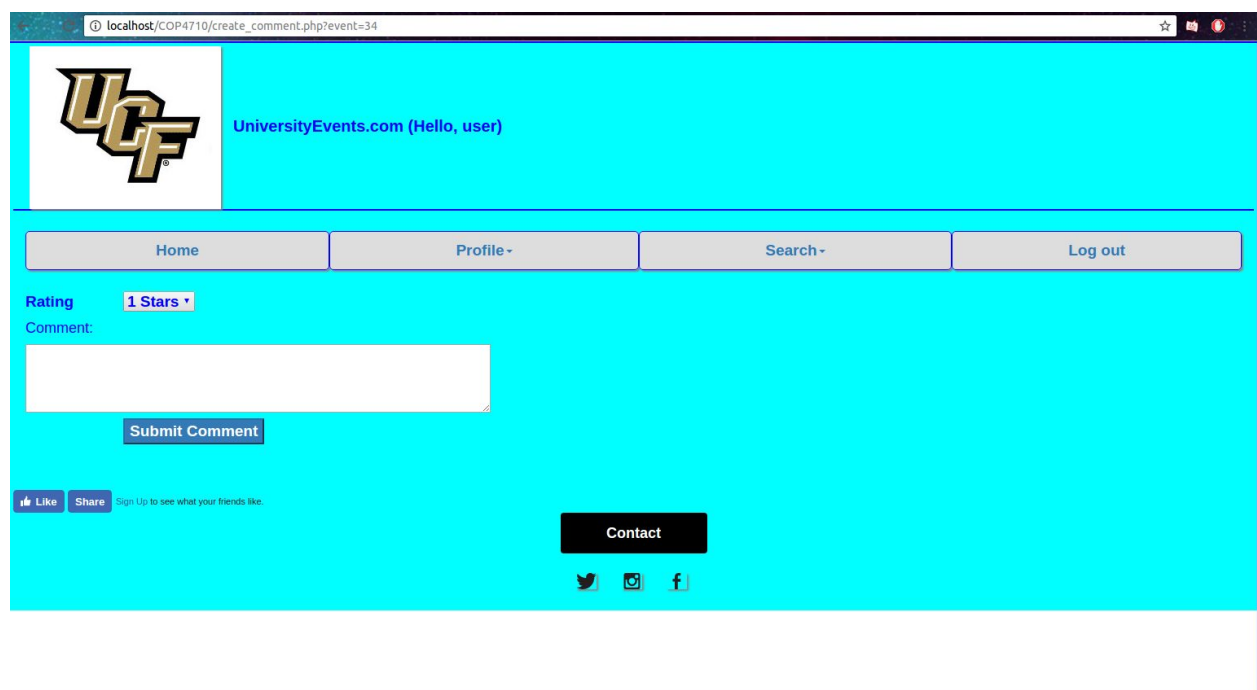
Twitter Instagram Facebook

Additionally, they are given a link (View Location on Map) which will lead them to a google maps page showing them the location of the event:



### Comment on event (Student only)

This is the comment page, which students can use to leave a comment and rating on whichever event they clicked the link from.



### Modify Comment (Student only)

This page is essentially the same as the previous page, except it will also show the contents of the previous comment that the user is currently trying to modify.



localhost/COP4710/update\_comment.php?event=34&date\_time=14th%20of%20April%202017%2012:44:48%20PM

UniversityEvents.com (Hello, user)

Home Profile - Search - Log out

Original Comment

user on 14th of April 2017 12:44:48 PM (3 stars)

ewewewewew

Rating 1 Stars

Comment:

ewewewewew

Submit Comment

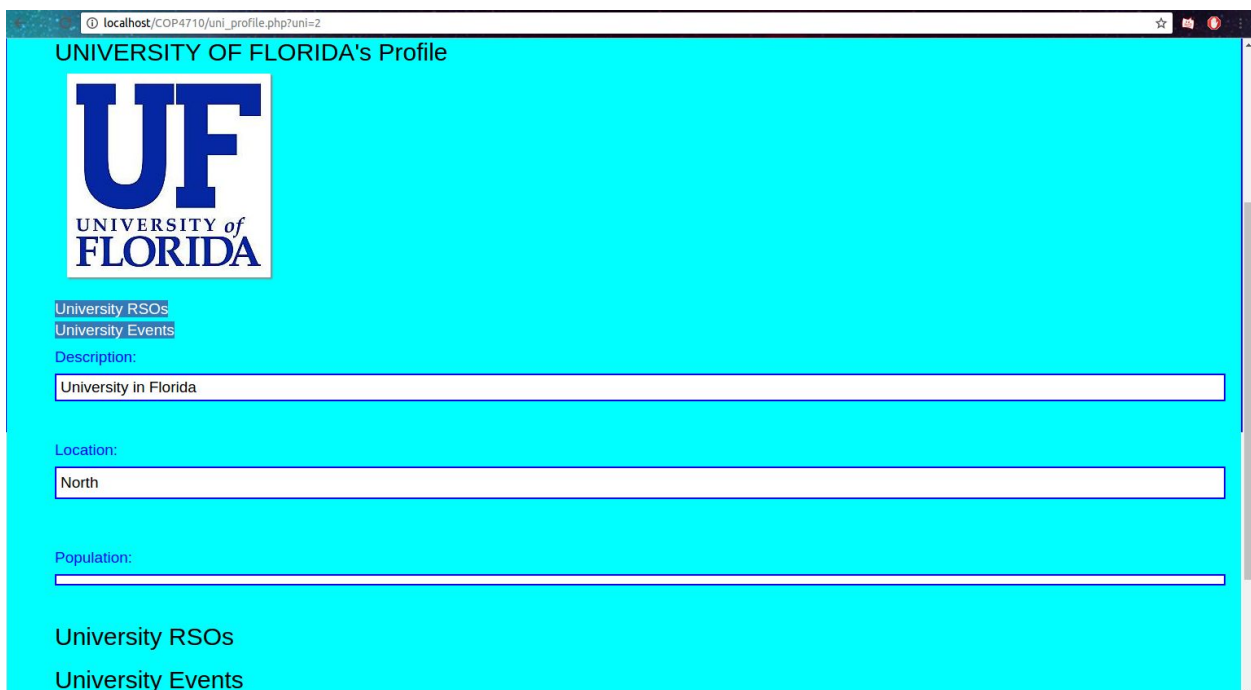
Like Share Sign Up to see what your friends like.

Contact

Twitter Instagram Facebook

### View University Profile

This is just a simple 'profile page' for the respective university.



localhost/COP4710/uni\_profile.php?uni=2

UNIVERSITY OF FLORIDA's Profile

UNIVERSITY of FLORIDA

University RSOs

University Events

Description:

University in Florida

Location:

North

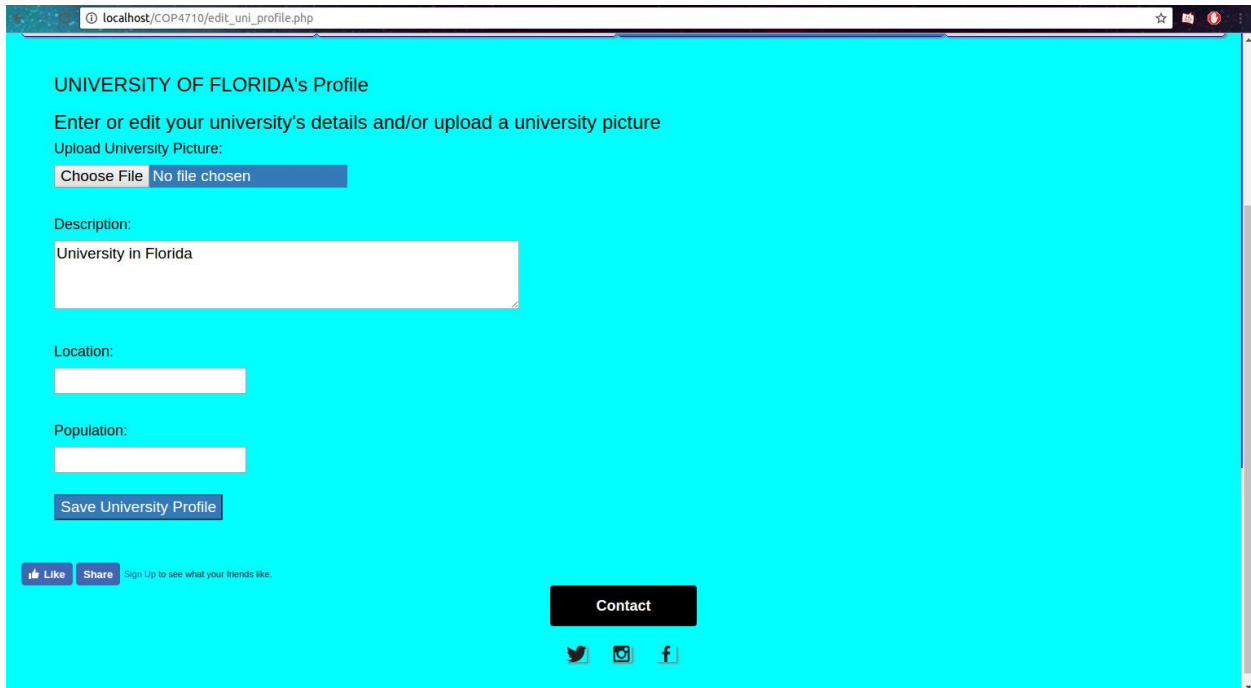
Population:

University RSOs

University Events

### *Edit University Profile (Superadmin only)*

This page is only accessible by the superadmin which created the university. It gives them the option to edit the basic information about the university.



The screenshot shows a web browser window with the address bar displaying 'localhost/COP4710/edit\_uni\_profile.php'. The page title is 'UNIVERSITY OF FLORIDA's Profile'. Below the title, there is a text prompt: 'Enter or edit your university's details and/or upload a university picture'. Underneath, the 'Upload University Picture:' section features a 'Choose File' button and a 'No file chosen' status. The 'Description:' section has a text input field containing 'University in Florida'. The 'Location:' section has an empty text input field. The 'Population:' section has an empty text input field. A 'Save University Profile' button is located below the population field. At the bottom left, there are 'Like' and 'Share' buttons with a small text link 'Sign Up to see what your friends like.' to the right. At the bottom center, there is a 'Contact' button. At the bottom right, there are three social media icons: Twitter, Instagram, and Facebook.

UNIVERSITY OF FLORIDA's Profile

Enter or edit your university's details and/or upload a university picture

Upload University Picture:

Choose File No file chosen

Description:

University in Florida

Location:

Population:

Save University Profile

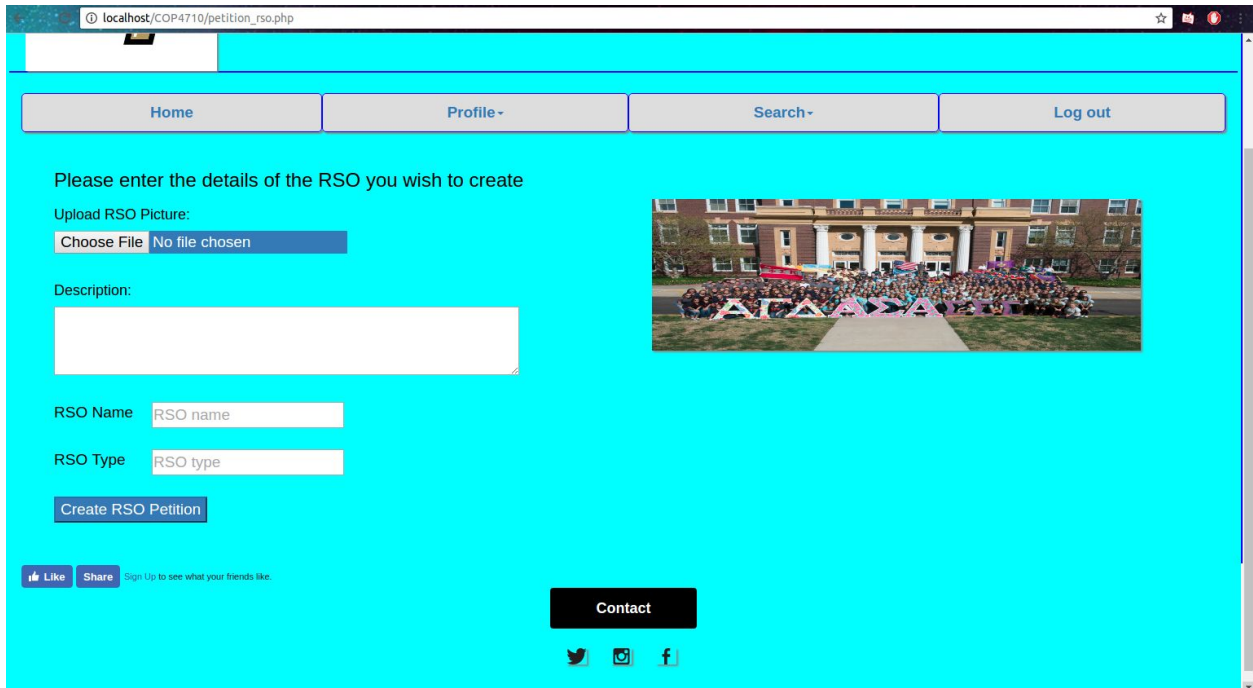
Like Share Sign Up to see what your friends like.

Contact

Twitter Instagram Facebook

### Create RSO Petition (Student only)

This page is only accessible to students, and it gives them the option to create the initial petition for a new RSO. From this, they are given a link which they can give to other students which will allow them to back the petition (if they meet the conditions to do so).



The screenshot shows a web browser window with the address bar displaying 'localhost/COP4710/petition\_rso.php'. The page has a light blue header with a navigation bar containing 'Home', 'Profile', 'Search', and 'Log out' buttons. Below the header, the main content area is white and contains the following elements:

- A heading: 'Please enter the details of the RSO you wish to create'.
- An 'Upload RSO Picture:' section with a 'Choose File' button and a 'No file chosen' status.
- A 'Description:' label followed by a large text input field.
- An 'RSO Name' label followed by a text input field containing the placeholder 'RSO name'.
- An 'RSO Type' label followed by a text input field containing the placeholder 'RSO type'.
- A 'Create RSO Petition' button.
- A social media section with 'Like' and 'Share' buttons, and a link to 'Sign Up to see what your friends like.'.
- A 'Contact' button.
- Social media icons for Twitter, Instagram, and Facebook.

On the right side of the form, there is a placeholder image showing a large group of students posing in front of a brick building.

### RSO Petition

As was discussed in the previous section, students that create the initial petition for an RSO are given a link which they can share with other students, which will lead them to this page. If they meet the constraints to back the RSO petition (same email domain as creator), they will automatically be added to the list of followers, and they will be included in the RSO member count. This page can also be reached if an inactive RSO (< 5 members) is clicked from the 'Search RSOs' page.

[Home](#)
[Profile -](#)
[Search -](#)
[Log out](#)

Please enter the details of the RSO you wish to create

Unofficial RSO created.


Once at least 5 students are following the RSO, it will become official. Direct them to [this page](#) (they must have the same email domain as you) Additionally, your account will be made an Admin account when the RSO becomes official.

Upload RSO Picture:

[Choose File](#) No file chosen

Description:

asdadsad



[Home](#)
[Profile -](#)
[Search -](#)
[Log out](#)



UniversityEvents.com (Hello, user)

## RSO Petition Page For asdsadasda

You have already petitioned this RSO. You may only do so once.

[Like](#)
[Share](#)

Sign Up to see what your friends like.

[Contact](#)

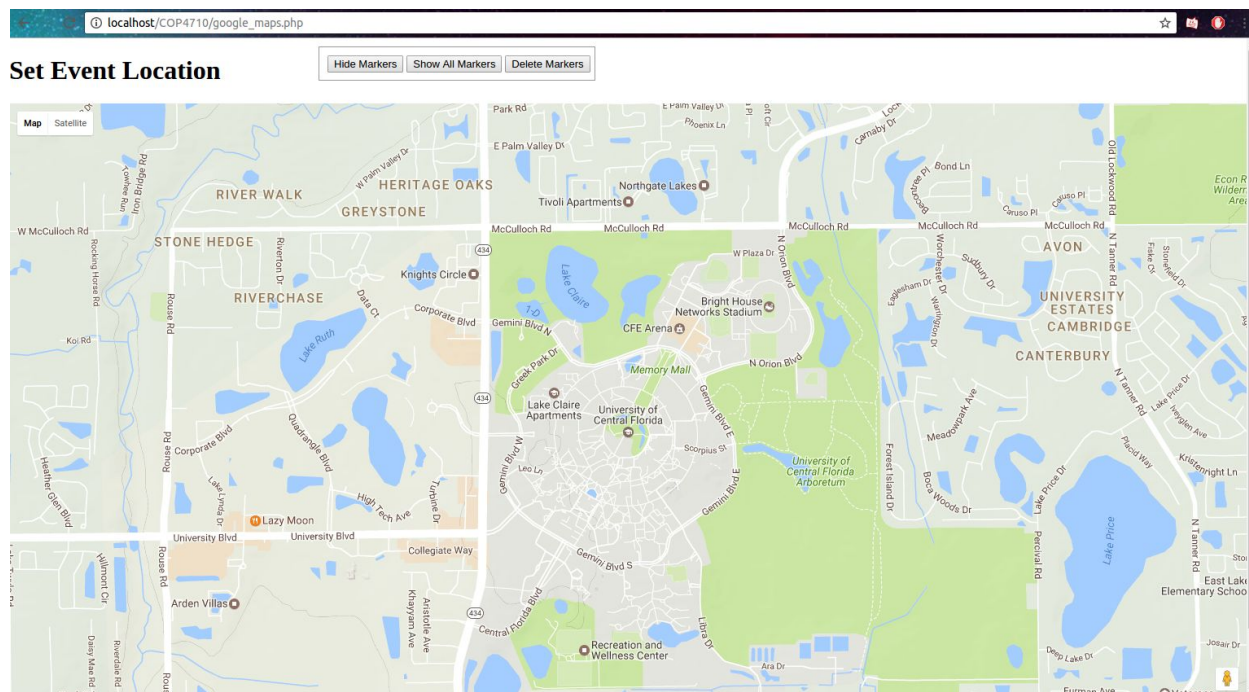






### Create Event (Admins only)

Before reaching the actual 'Create Event' page, admins are sent to a google maps page, which they must use to set the location of the event - they would simply click on the map once, wherever the event is to be held.



From there, they will be lead to the actual event creation page, on which they can input the information regarding the event.

Please enter the details of the Event you wish to create

Description:

RSO Name:

Category:

Location:

Date:

Start time:  ☐ AM ☒ PM

End time:  ☐ AM ☒ PM

Contact Name:

Contact:

Contact Phone:

Contact Email:

University:

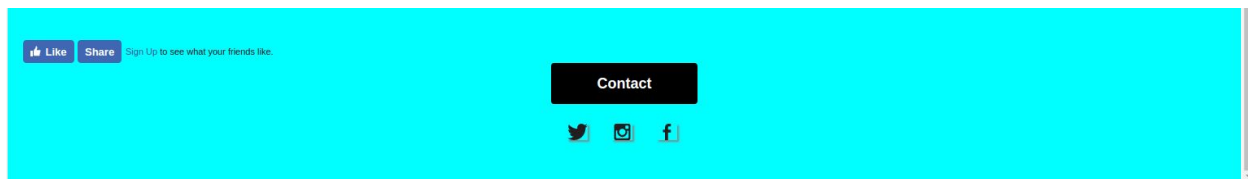
Event Scope: ☒ Public ☐ Private

Max Occupancy:

Create Event

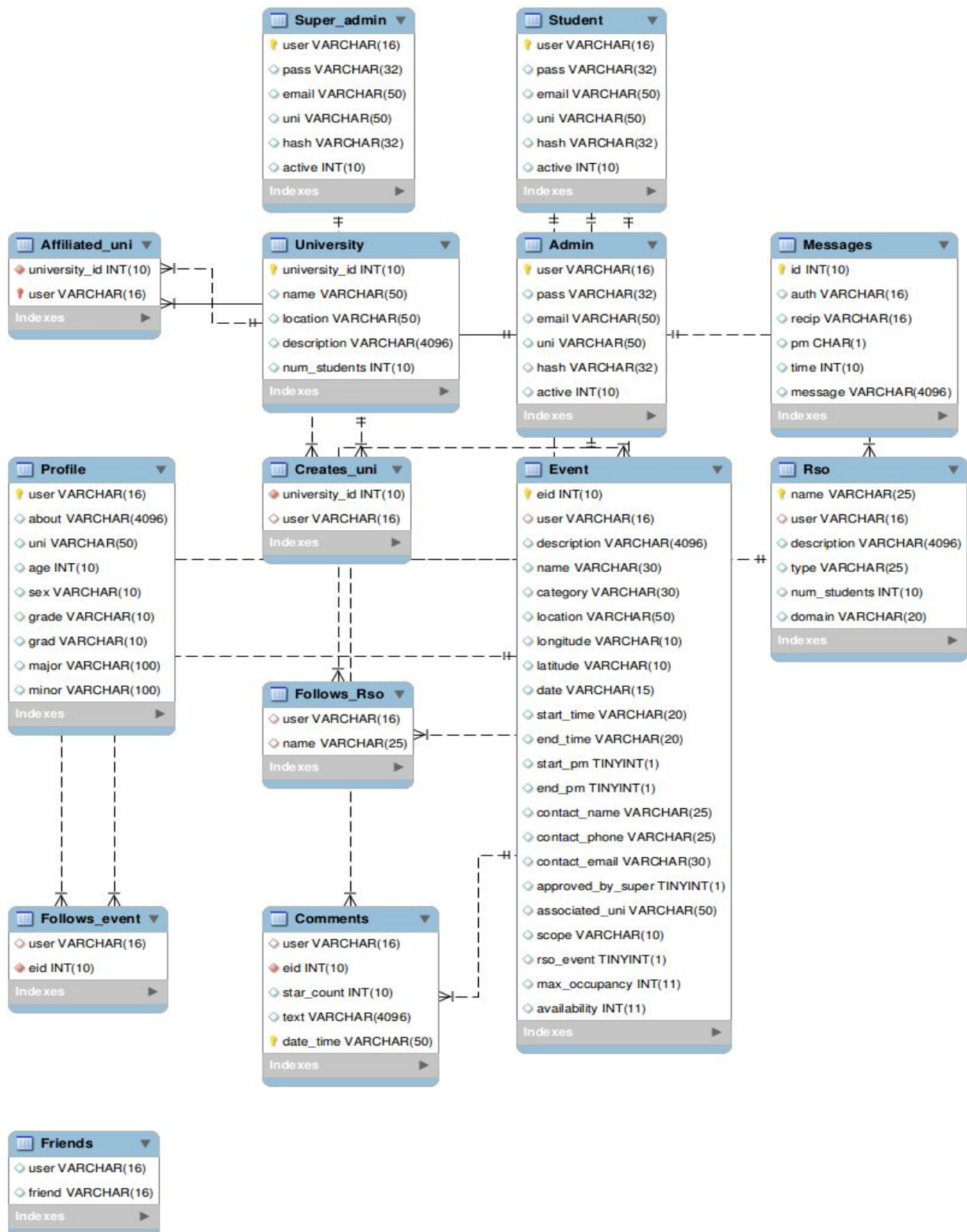
### *Footer (Facebook int., Contact)*

The footer for the site, which is found on every page, is very simple and consists of only a few components. Firstly, it fulfills the social media integration requirement of the project by providing users with a link to like, follow, and post about the site (if it had a facebook page, which it does not) and to sign up to facebook. Additionally, it has a 'Contact' button, which opens up an email service, and 3 buttons with links to the site's social media accounts on Twitter, Instagram, and Facebook (although, it does not have any).





## ER-Model



The above EER diagram does not provide any actual enforcement of project-specific constraints (apart from those enforced through the use of keys). In the back-end code, however, (in PHP) the project constraints are enforced, completely, through the use of conditional statements that check the results of queries made to the database. Though this is not as optimal in terms of performance - we are making more queries in order to check for the same thing - but it is simpler to implement, which is the reason I decided to go with this approach.

In fact, a little more than halfway through implementation, I hand-wrote a proper ER diagram for the database that would allow the server to handle the constraints for me, however by that time too much was done in the PHP to justify making the change. Nonetheless, rest assured that the constraints of the project are all handled and enforced, one way or another.

### **Database Schema**

```
DELETE DATABASE IF EXISTS UniEvents;
```

```
CREATE DATABASE UniEvents;
```

```
CREATE USER 'COP4710'@'localhost' IDENTIFIED BY 'password';
```

```
GRANT ALL PRIVILEGES ON UniEvents.* TO 'COP4710'@'localhost';
```

```
FLUSH PRIVILEGES;
```

```
USE UniEvents;
```

```
CREATE TABLE Super_admin VALUES(
    user VARCHAR(16) PRIMARY KEY,
    pass VARCHAR(32),
    email VARCHAR(50),
    uni VARCHAR(50),
    hash VARCHAR(32),
    active INT UNSIGNED,
    INDEX(user(6)));
```

```
CREATE TABLE Admin VALUES(
    user VARCHAR(16) PRIMARY KEY,
```

```

pass VARCHAR(32),
email VARCHAR(50),
uni VARCHAR(50),
hash VARCHAR(32),
active INT UNSIGNED,
INDEX(user(6)));

```

```

CREATE TABLE Student VALUES(
    user VARCHAR(16) PRIMARY KEY,
    pass VARCHAR(32),
    email VARCHAR(50),
    uni VARCHAR(50),
    hash VARCHAR(32),
    active INT UNSIGNED,
    INDEX(user(6)));

```

```

CREATE TABLE Profile VALUES(
    user VARCHAR(16) PRIMARY KEY,
    about VARCHAR(4096),
    uni VARCHAR(50),
    age INT UNSIGNED,
    sex VARCHAR(10),
    grade VARCHAR(10),
    grad VARCHAR(10),
    major VARCHAR(100),
    minor VARCHAR(100),
    INDEX(user(6)));

```

```

CREATE TABLE University VALUES(
    university_id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50),
    location VARCHAR(50),
    description VARCHAR(4096),
    num_students INT UNSIGNED,
    INDEX(name(6)),
    INDEX(university_id));

```

```

CREATE TABLE Creates_uni VALUES(
    university_id INT UNSIGNED AUTO_INCREMENT NOT NULL,

```

```

user VARCHAR(16),
FOREIGN KEY (university_id) REFERENCES University(university_id),
FOREIGN KEY (user) REFERENCES Super_admin(user));

```

```

CREATE TABLE Affiliated_uni VALUES(
    university_id INT UNSIGNED AUTO_INCREMENT NOT NULL,
    user VARCHAR(16) PRIMARY KEY,
    FOREIGN KEY (university_id) REFERENCES University(university_id),
    FOREIGN KEY (user) REFERENCES Admin(user));

```

```

CREATE TABLE Event VALUES(
    eid INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    user VARCHAR(16),
    description VARCHAR(4096),
    name VARCHAR(30),
    category VARCHAR(30),
    location VARCHAR(50),
    longitude VARCHAR(10),
    latitude VARCHAR(10),
    date VARCHAR(15),
    start_time VARCHAR(20),
    end_time VARCHAR(20),
    start_pm BOOLEAN,
    end_pm BOOLEAN,
    contact_name VARCHAR(25),
    contact_phone VARCHAR(25),
    contact_email VARCHAR(30),
    approved_by_super BOOLEAN,
    associated_uni VARCHAR(50),
    scope VARCHAR(10),
    rso_event BOOLEAN,
    max_occupancy INT,
    availability INT,
    FOREIGN KEY (user) REFERENCES Admin (user) ON DELETE CASCADE);

```

```

CREATE TABLE Comments VALUES(
    user VARCHAR(16),
    eid INT UNSIGNED AUTO_INCREMENT,
    star_count VARCHAR(1),

```

```

text VARCHAR(4096),
date_time VARCHAR(50) PRIMARY KEY,
FOREIGN KEY (user) REFERENCES Student (user),
FOREIGN KEY (eid) REFERENCES Event (eid));

```

```

CREATE TABLE Follows_event VALUES(
    user VARCHAR(16),
    eid INT UNSIGNED AUTO_INCREMENT,
    FOREIGN KEY (user) REFERENCES Student (user),
    FOREIGN KEY (eid) REFERENCES Event (eid));

```

```

CREATE TABLE Rso VALUES(
    name VARCHAR(25) PRIMARY KEY,
    user VARCHAR(16),
    description VARCHAR(4096),
    type VARCHAR(25),
    num_students INT UNSIGNED,
    domain VARCHAR(20),
    FOREIGN KEY (user) REFERENCES Admin (user) ON DELETE CASCADE);

```

```

CREATE TABLE Follows_Rso VALUES(
    user VARCHAR(16),
    name VARCHAR(25),
    FOREIGN KEY (user) REFERENCES Student (user) ON DELETE CASCADE,
    FOREIGN KEY (name) REFERENCES Rso (name));

```

### Sample Data Population

What follows is the PHP code used to populate the database with sample data. It includes:

- 3 superadmins, and therefore 3 universities
- 3 admins, and therefore 3 RSOs
- 17 students
- 5 events
- 4 comments (5 in total, one was updated then deleted)

```

function queryMysql($query)
{
    global $connection;

```

```

$result = $connection->query($query);
if (!$result) die($connection->error);
return $result;
}

```

```

queryMysql("INSERT INTO Super_admin VALUES(
    'Superadmin1',
    '5f4dcc3b5aa765d61d8327deb882cf99',
    'Superadmin1@ucf.edu',
    'university of Central florida',
    'hash',
    '1')");

```

```

queryMysql("INSERT INTO Profile VALUES('Superadmin1', NULL, 'university of central florida',
NULL, NULL, NULL, NULL, NULL, NULL));

```

```

queryMysql("INSERT INTO University VALUES(NULL, 'university of central florida', NULL, NULL,
NULL));

```

```

queryMysql("INSERT INTO Creates_uni VALUES('1', 'Superadmin1')");

```

```

queryMysql("INSERT INTO Super_admin VALUES(
    'Superadmin2',
    '5f4dcc3b5aa765d61d8327deb882cf99',
    'Superadmin2@uf.edu',
    'university of florida',
    'hash',
    '1')");

```

```

queryMysql("INSERT INTO Profile VALUES('Superadmin2', NULL, 'university of florida', NULL,
NULL, NULL, NULL, NULL, NULL));

```

```

queryMysql("INSERT INTO University VALUES(NULL, 'university of florida', NULL, NULL,
NULL));

```

```

queryMysql("INSERT INTO Creates_uni VALUES('2', 'Superadmin2')");

```

```

queryMysql("INSERT INTO Super_admin VALUES(

```

```
'Superadmin3',
'5f4dcc3b5aa765d61d8327deb882cf99',
'Superadmin3@fsu.edu',
'florida state university',
'hash',
'1')");
```

```
queryMysql("INSERT INTO Profile VALUES('Superadmin3', NULL, 'florida state university',
NULL, NULL, NULL, NULL, NULL, NULL));
```

```
queryMysql("INSERT INTO University VALUES(NULL, 'florida state university', NULL, NULL,
NULL));
```

```
queryMysql("INSERT INTO Creates_uni VALUES('3', 'Superadmin3')");
```

```
for ($i = 1; $i <= 10; $i++)
{
    queryMysql("INSERT INTO Student VALUES('student$i',
'5f4dcc3b5aa765d61d8327deb882cf99', 'student$i@knights.ucf.edu', 'university of central
florida', 'hash', '1')");
}
for ($i = 11; $i <= 15; $i++)
{
    queryMysql("INSERT INTO Student VALUES('student$i',
'5f4dcc3b5aa765d61d8327deb882cf99', 'student$i@uf.edu', 'university of florida', 'hash', '1')");
}
for ($i = 16; $i <= 20; $i++)
{
    queryMysql("INSERT INTO Student VALUES('student$i',
'5f4dcc3b5aa765d61d8327deb882cf99', 'student$i@fsu.edu', 'florida state university', 'hash',
'1')");
}
```

```
queryMysql("INSERT INTO Admin VALUES('student1', '5f4dcc3b5aa765d61d8327deb882cf99',
'student1@knights.ucf.edu', 'university of central florida', 'hash', '0')");
```

```
queryMysql("INSERT INTO Rso VALUES('ucfRSO', 'student1', 'RSO for UCF', 'type1', '1',
'knights.ucf.edu')");
```

```
queryMysql("INSERT INTO Follows_Rso VALUES('student1', 'ucfRSO')");
```

```
for ($i = 2; $i <= 5; $i++)
```

```
{
```

```
    queryMysql("UPDATE Rso SET num_students = num_students + 1 WHERE name='ucfRSO'");
```

```
    queryMysql("INSERT INTO Follows_Rso VALUES('student$i', 'ucfRSO')");
```

```
}
```

```
queryMysql("DELETE FROM Follows_Rso WHERE user='student1'");
```

```
queryMysql("DELETE FROM Student WHERE user='student1'");
```

```
queryMysql("UPDATE Admin SET active = 1 WHERE user='student1'");
```

```
queryMysql("INSERT INTO Admin VALUES('student11',  
'5f4dcc3b5aa765d61d8327deb882cf99', 'student11@uf.edu', 'university of florida', 'hash',  
'0')");
```

```
queryMysql("INSERT INTO Rso VALUES('ufRSO', 'student11', 'RSO for UF', 'type2', '1',  
'uf.edu')");
```

```
queryMysql("INSERT INTO Follows_Rso VALUES('student11', 'ufRSO')");
```

```
for ($i = 12; $i <= 15; $i++)
```

```
{
```

```
    queryMysql("UPDATE Rso SET num_students = num_students + 1 WHERE name='ufRSO'");
```

```
    queryMysql("INSERT INTO Follows_Rso VALUES('student$i', 'ufRSO')");
```

```
}
```

```
queryMysql("DELETE FROM Follows_Rso WHERE user='student11'");
```

```
queryMysql("DELETE FROM Student WHERE user='student11'");
```

```
queryMysql("UPDATE Admin SET active = 1 WHERE user='student11'");
```

```
queryMysql("INSERT INTO Admin VALUES('student16',  
'5f4dcc3b5aa765d61d8327deb882cf99', 'student16@fsu.edu', 'florida state university', 'hash',  
'0')");
```

```
queryMysql("INSERT INTO Rso VALUES('fsuRSO', 'student16', 'RSO for FSU', 'type3', '1',  
'fsu.edu')");
```

```
queryMysql("INSERT INTO Follows_Rso VALUES('student16', 'fsuRSO')");
```



```

for ($i = 17; $i <= 20; $i++)
{
    queryMysql("UPDATE Rso SET num_students = num_students + 1 WHERE name='fsuRSO'");
    queryMysql("INSERT INTO Follows_Rso VALUES('student$i', 'fsuRSO')");
}

```

```

queryMysql("DELETE FROM Follows_Rso WHERE user='student16'");
queryMysql("DELETE FROM Student WHERE user='student16'");
queryMysql("UPDATE Admin SET active = 1 WHERE user='student16'");

```

```

queryMysql("INSERT INTO Event VALUES( NULL, 'student1', 'description of event1', 'event1',
'category1',
    'Student Union', '-81.2004', '28.6018', '04/18/17','1400', '1500', '1',
    '1', 'Guillermo', '1234512345', 'student1@knights.ucf.edu', FALSE, 'university of central
florida',
    'Public', '0', '100', '100')");

```

```

queryMysql("INSERT INTO Event VALUES( NULL, 'student1', 'description of event1', 'event1',
'category1',
    'Student Union', '-81.2004', '28.6018', '04/18/17','1500', '1600', '1',
    '1', 'Guillermo', '1234512345', 'student1@knights.ucf.edu', FALSE, 'university of central
florida',
    'Public', '0', '100', '100')");

```

```

queryMysql("INSERT INTO Event VALUES( NULL, 'student1', 'description of event1', 'event1',
'category1',
    'Student Union', '-81.2004', '28.6018', '04/18/17','1600', '1700', '1',
    '1', 'Guillermo', '1234512345', 'student1@knights.ucf.edu', FALSE, 'university of central
florida',
    'Public', '0', '100', '100')");

```

```

queryMysql("INSERT INTO Event VALUES( NULL, 'student1', 'description of event2', 'event2',
'category2',
    'Student Union', '-81.2004', '28.6018', '04/20/17','0800', '0900', '0',
    '0', 'Guillermo', '1234512345', 'student1@knights.ucf.edu', FALSE, 'university of central
florida',
    'RSO', '1', '200', '200')");

```

```
queryMysql("INSERT INTO Event VALUES( NULL, 'student1', 'description of event3', 'event3',
'category3',
'HEC', '-82.3449', '29.6476', '04/20/17', '1000', '1100', '0',
'0', 'Guillermo', '1234512345', 'student1@knights.ucf.edu', FALSE, 'university of central
florida',
'Private', '0', '50', '50')");
```

```
queryMysql("INSERT INTO Event VALUES( NULL, 'student11', 'description of event4', 'event4',
'category4',
'Science Library', '-81.1974', '28.6005', '04/20/17', '1000', '1100', '0',
'0', 'Guillermo2', '1234512345', 'student11@uf.edu', FALSE, 'university of florida',
'Private', '0', '50', '50')");
```

```
queryMysql("INSERT INTO Event VALUES( NULL, 'student16', 'description of event5', 'event5',
'category5',
'Ruby Diamond Concert Hall', '-84.2918', '30.4405', '04/20/17', '1000', '1100', '0',
'0', 'Guillermo3', '1234512345', 'student1@knights.ucf.edu', FALSE, 'university of central
florida',
'Public', '0', '50', '50')");
```

```
$date = date('jS \of F Y h:i:s A');
queryMysql("INSERT INTO Comments VALUES('student3', '1', '5', 'This will be an awesome
event!', '$date')");
```

```
sleep(2);
$date = date('jS \of F Y h:i:s A');
queryMysql("INSERT INTO Comments VALUES('student7', '1', '1', 'This is going to be terrible.',
'$date')");
```

```
sleep(2);
$date = date('jS \of F Y h:i:s A');
queryMysql("INSERT INTO Comments VALUES('student13', '4', '3', 'This will be an ok event.',
'$date')");
```

```
sleep(2);
$date = date('jS \of F Y h:i:s A');
queryMysql("INSERT INTO Comments VALUES('student17', '5', '5', 'My opinion will change
shortly.', '$date')");
```

```

sleep(2);
queryMysql("DELETE FROM Comments WHERE eid='5' AND user='student16' AND
date_time='$date'");

$date2 = date('jS \of F Y h:i:s A');
queryMysql("INSERT INTO Comments VALUES('student17', '5', '1', 'My opinion has changed!',
'$date2')");

queryMysql("DELETE FROM Comments WHERE eid='5' AND user='student17' AND
date_time='$date'");

?>

<br>
</body>
</html>

```

### Results

The results of this code running - after the database has already been created and a connection has been made with it - is included below:

#### Superadmin Table

```

mysql> mysql> select * from Super_admin;
+-----+-----+-----+-----+-----+-----+
| user      | pass                                     | email                               | uni                                | hash | active |
+-----+-----+-----+-----+-----+-----+
| Superadmin1 | 5f4dcc3b5aa765d61d8327deb882cf99 | Superadmin1@ucf.edu | university of Central florida | hash | 1 |
| Superadmin2 | 5f4dcc3b5aa765d61d8327deb882cf99 | Superadmin2@uf.edu | university of florida | hash | 1 |
| Superadmin3 | 5f4dcc3b5aa765d61d8327deb882cf99 | Superadmin3@fsu.edu | florida state university | hash | 1 |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

```

#### Admin Table

```

mysql> select * from Admin;
+-----+-----+-----+-----+-----+-----+
| user      | pass                                     | email                               | uni                                | hash | active |
+-----+-----+-----+-----+-----+-----+
| student1  | 5f4dcc3b5aa765d61d8327deb882cf99 | student1@knights.ucf.edu | university of central florida | hash | 1 |
| student11 | 5f4dcc3b5aa765d61d8327deb882cf99 | student11@uf.edu | university of florida | hash | 1 |
| student16 | 5f4dcc3b5aa765d61d8327deb882cf99 | student16@fsu.edu | florida state university | hash | 1 |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

```

Student Table

```
mysql> select * from Student;
```

user	pass	email	uni	hash	active
student10	5f4dcc3b5aa765d61d8327deb882cf99	student10@knights.ucf.edu	university of central florida	hash	1
student12	5f4dcc3b5aa765d61d8327deb882cf99	student12@uf.edu	university of florida	hash	1
student13	5f4dcc3b5aa765d61d8327deb882cf99	student13@uf.edu	university of florida	hash	1
student14	5f4dcc3b5aa765d61d8327deb882cf99	student14@uf.edu	university of florida	hash	1
student15	5f4dcc3b5aa765d61d8327deb882cf99	student15@uf.edu	university of florida	hash	1
student17	5f4dcc3b5aa765d61d8327deb882cf99	student17@fsu.edu	florida state university	hash	1
student18	5f4dcc3b5aa765d61d8327deb882cf99	student18@fsu.edu	florida state university	hash	1
student19	5f4dcc3b5aa765d61d8327deb882cf99	student19@fsu.edu	florida state university	hash	1
student2	5f4dcc3b5aa765d61d8327deb882cf99	student2@knights.ucf.edu	university of central florida	hash	1
student20	5f4dcc3b5aa765d61d8327deb882cf99	student20@fsu.edu	florida state university	hash	1
student3	5f4dcc3b5aa765d61d8327deb882cf99	student3@knights.ucf.edu	university of central florida	hash	1
student4	5f4dcc3b5aa765d61d8327deb882cf99	student4@knights.ucf.edu	university of central florida	hash	1
student5	5f4dcc3b5aa765d61d8327deb882cf99	student5@knights.ucf.edu	university of central florida	hash	1
student6	5f4dcc3b5aa765d61d8327deb882cf99	student6@knights.ucf.edu	university of central florida	hash	1
student7	5f4dcc3b5aa765d61d8327deb882cf99	student7@knights.ucf.edu	university of central florida	hash	1
student8	5f4dcc3b5aa765d61d8327deb882cf99	student8@knights.ucf.edu	university of central florida	hash	1
student9	5f4dcc3b5aa765d61d8327deb882cf99	student9@knights.ucf.edu	university of central florida	hash	1

17 rows in set (0.00 sec)

University Table

```
mysql> select * from University;
```

university_id	name	location	description	num_students
1	university of central florida	NULL	NULL	NULL
2	university of florida	NULL	NULL	NULL
3	florida state university	NULL	NULL	NULL

3 rows in set (0.00 sec)

RSO Table

```
mysql> select * from Rso;
```

name	user	description	type	num_students	domain
fsuRSO	student16	RSO for FSU	type3	5	fsu.edu
ucfRSO	student1	RSO for UCF	type1	5	knights.ucf.edu
ufRSO	student11	RSO for UF	type2	5	uf.edu

3 rows in set (0.00 sec)

Follows\_Rso Table

```
mysql> select * from Follows_Rso;
```

user	name
student2	ucfRSO
student3	ucfRSO
student4	ucfRSO
student5	ucfRSO
student12	ufRSO
student13	ufRSO
student14	ufRSO
student15	ufRSO
student17	fsuRSO
student18	fsuRSO
student19	fsuRSO
student20	fsuRSO

```
12 rows in set (0.00 sec)
```

Event Table

```
mysql> select * from Event;
```

eid	user	description	name	category	location	longitude	latitude	date	start_time	end_time	start_pn	end_pn
1	student1	description of event1	event1	category1	Student Union	-81.2004	28.6018	04/18/17	1400	1500	1	1
2	Guillermo	1234512345	student1@knights.ucf.edu	0	university of central florida	-81.2004	28.6018	04/18/17	1500	1600	1	1
3	Guillermo	1234512345	student1@knights.ucf.edu	0	university of central florida	-81.2004	28.6018	04/18/17	1600	1700	1	1
4	Guillermo	1234512345	student1@knights.ucf.edu	0	university of central florida	-81.2004	28.6018	04/20/17	0800	0900	0	0
5	Guillermo	1234512345	student1@knights.ucf.edu	0	university of central florida	-82.3449	29.6476	04/20/17	1000	1100	0	0
6	Guillermo2	1234512345	student11@uf.edu	0	university of central florida	-81.1974	28.6005	04/20/17	1000	1100	0	0
7	Guillermo3	1234512345	student1@knights.ucf.edu	0	university of central florida	-84.2918	30.4405	04/20/17	1000	1100	0	0

```
7 rows in set (0.00 sec)
```

Comment Table



```
mysql> select * from Comments;
```

user	eid	star_count	text	date_time
student3	1	5	This will be an awesome event!	14th of April 2017 01:39:06 PM
student7	1	1	This is going to be terrible.	14th of April 2017 01:39:08 PM
student13	4	3	This will be an ok event.	14th of April 2017 01:39:11 PM
student17	5	1	My opinion has changed!	14th of April 2017 01:39:15 PM
student7	1	1	This is going to be terrible.	14th of April 2017 01:53:07 PM

```
5 rows in set (0.00 sec)
```

### Examples of SQL Queries Used

/\* SQL Statements relating to RSOs:

```

**
** --Queries once a student has initially petitioned an RSO--
** queryMysql("UPDATE Admin SET active = 0 WHERE user='$user'");
** queryMysql("INSERT INTO Rso VALUES('$name', '$user', '$description', '$type',
** '$num_students', '$domain')");
** queryMysql("INSERT INTO Follows_Rso VALUES('$user', '$name')");
**
** --Queries for each subsequent student that petitions RSO (num_students < 5)--
** queryMysql("UPDATE Rso SET num_students = num_students + 1 WHERE name='$name'");
** queryMysql("INSERT INTO Follows_Rso VALUES('$user', '$name')");
**
** --Queries once the 5th student has petitioned RSO (num_students == 5)--
** queryMysql("DELETE FROM Follows_Rso WHERE user='$admin'");
** queryMysql("DELETE FROM Student WHERE user='$admin'");
** queryMysql("UPDATE Admin SET active = 1 WHERE user='$admin'");
**
** --Queries for a student to follow an active RSO--
** queryMysql("INSERT INTO Follows_Rso VALUES('$user', '$rso')");
** queryMysql("UPDATE Rso SET num_students = num_students + 1 WHERE name='$rso'");
**
** --Queries when an active RSO drops to below 5 students--
** queryMysql("UPDATE Admin SET active = '0' WHERE user='$admin'");
** queryMysql("INSERT INTO Student(user, pass, email, uni, hash, active) SELECT * FROM
Admin ** WHERE user = '$admin'");
** queryMysql("INSERT INTO Follows_Rso VALUES('$admin', '$rso')");
*/

```

/\* SQL Statements relating to Events:

```

**
** --Queries to create new Event (a unique event is added for every hr block that an event
takes ** up)--
** for ($i = 0; $i < $chunk; $i+=100)
**{
** $time = $start_time + $i;
** queryMysql("INSERT INTO Event VALUES( NULL, '$user', '$description', '$name', '$category',
**      '$location', '$longitude', '$latitude', '$date', '$time', '$end_time', '$start_pm',
**      '$end_pm', '$contact_name', '$contact_phone', '$contact_email', FALSE,
** '$associated_uni', '$scope', '$rso_event', '$max_occupancy', '$max_occupancy')");
**}
**
** --Queries to follow an existing event (assuming they meet all constraints to do so)--
** queryMysql("INSERT INTO Follows_event VALUES('$user', '$event')");
** queryMysql("UPDATE Event SET availability = availability - 1 WHERE eid='$event'");
*/

/* SQL Statements relating to Comments (assuming constraints have passed):
**
** --Queries to create a comment--
** queryMysql("INSERT INTO Comments VALUES('$user', '$eid', '$rating', '$text', '$date')");
**
** --Queries to modify a comment--
** queryMysql("DELETE FROM Comments WHERE eid='$eid' AND user='$user' AND
** date_time='$date_time'");
** queryMysql("INSERT INTO Comments VALUES('$user', '$eid', '$rating', '$text', '$date')");
**
** --Queries to delete a comment--
** queryMysql("DELETE FROM Comments WHERE eid='$eid' AND user='$user' AND
** date_time='$date_time'");
*/

/* SQL Statements relating to searches:
**
** --Queries to search for an event by University only--
** $event = queryMysql("SELECT * FROM Event WHERE (rso_event='1' AND
** associated_uni='$uni' AND location='$location')
**      OR (approved_by_super='1' AND associated_uni='$uni' AND location='$location')
**      OR (scope='Public' AND associated_uni='$uni' AND location='$location')");

```

```

**
** --Queries to search for an event by University and Location--
** $event = queryMysql("SELECT * FROM Event WHERE (rso_event='1' AND
** associated_uni='$uni')
**          OR (approved_by_super='1' AND associated_uni='$uni')
**          OR (scope='Public' AND associated_uni='$uni')");
**
** --Queries to check for RSO Constraint--
** if ($rso_event)
** {
**   $admin = $event['user'];
**   $rso = queryMysql("SELECT * FROM Rso WHERE user='$admin'");
**   $rso = $rso->fetch_array();
**   $rso = $rso['name'];
**   $follows = queryMysql("SELECT * FROM Follows_Rso WHERE user='$user' AND
** name='$rso'");
**   if (!$follows->num_rows)
**     continue;
** }
**
** --Check if student is in University if scope is private--
** elseif ($scope == "Private")
** {
**   $user_uni = $result['uni'];
**   if ($user_uni != $uni)
**     continue;
** }
**
** --Queries to search for an RSO by University only--
** $rso = queryMysql("SELECT * FROM Rso WHERE user=(SELECT user FROM Admin WHERE
** uni='$uni')");
**
** --Queries to search for an RSO by University and Location--
** $rso = queryMysql("SELECT * FROM Rso WHERE user=(SELECT user FROM Admin WHERE
** uni='$uni') AND type='$category'");
**
** --Query to ensure that student attempting to follow RSO attends RSO's university--
** queryMysql("SELECT * FROM Student WHERE user='$user' AND
**            uni=(SELECT uni FROM Admin WHERE

```



```
**      user=(SELECT user FROM Rso WHERE name='$rso'))");
*/
```

## How-to: Installing the Software

### Setting up Environment/Server

Before beginning to attempt to set up the system, you will need to install the LAMP stack, which consists of Apache server 2.0, MySQL, and PHP. Once this has been installed, you will need to locate your server directory (`~/var/www/html`) and place the entire source directory of the system in here.

Next, you will likely need to set the proper permissions to access the directory through Apache, since by default it has root privileges only. To grant a user privileges to properly use the system, you can use:

```
sudo chown -R <user>:<user> /var/www/html/COP4710
```

Next, before even attempting to navigate to a page, make sure that Apache server is up and running, and started. This can be done simply with:

```
sudo service apache2 restart
```

### Setting up Database

Next, you must properly create the database. Doing so is simple with the provided `database.sql` file, you can do it by following the steps below:

1. Open Mysql from Command Line
2. Type `source /var/www/html/database.sql` and `Enter`

If you happen to get an 'ERROR 1819', simply type 'uninstall plugin validate\_password;' and press ENTER (you should be logged in as root). Then should follow the steps again.

Finally, you are ready to start the application - all you have to do is open a compatible browser (development was done using Google Chrome), and type in

'localhost/COP4710/index.php' and that will lead you to the home page, where you can begin to explore the site.

## **Observations**

### Desired Features

Although all of the required functionality for the system is present, there were a few features I would have liked to be able to include, which I didn't.

1. I wanted to create a social network aspect to the application, so I would have liked to be able to include a messaging/friends feature.
2. I thought it would be a good idea to add more interactivity in the RSO page, so I wanted to implement a chat room feature for members of the same RSO to chat.
3. I would have definitely liked to have had a modern GUI, or at least more stylish and formatted better than the state in which it's in now.
4. I implemented the system using a procedural-style of programming, and I would have liked to have done it using an Object-Oriented style instead.
5. Better integration of Google Maps would have been nice as well. The current integration is awkward and was only done the way it was due to lack of time/commitments to other projects.

### Problems Encountered

Throughout the development of the system, I did not really encounter too many issues worth noting as problematic. I did encounter a few issues, however, that I wish would have been resolved or worked-around a bit quicker than it was. They include:

1. Integration of Google Maps was a pain for some reason - I could not get the map to display as part of its own 'small' div in the actual event creation page.
2. Initially, starting up the server to allow me to connect to the directory from the browser was an issue. I tried a few work-arounds before realizing that I could just change the permissions on the directory.

3. Some minute, specific requirements/constraints that were easily missed caused me to go back and try to fit it neatly in the code - something that proved to be a bit awkward and unnatural for whatever reason.

### Database Performance

It does not really seem necessary to evaluate query performance at this stage, simply because the queries - even if they are extremely poorly optimized - will always return fairly quickly, because the tables are so small. Therefore, I will forego including query evaluation times, as I feel they will not be useful. However, I will reiterate that the database could have been better - the scalability of query performance is not very good in the project as-is, and thus an improved layout of the database would be most beneficial, as it would 1) reduce the need for additional queries as a means of constraint enforcement and 2) it would allow the dbms optimizer to better perform its job.

### **Conclusion**

In conclusion, this was a very fun project to work on and I am quite pleased with the outcome. As was said earlier, I believe I could have done a better job with the GUI and structure of the database, but all things considered (I had to learn PHP, Javascript, and SQL), I think the result was not bad.