Table of Contents

**Table of Contents1**

**Project Description**

**ER Diagram1**

**Front End Code2**

**API Design3**

**Database4**

**Code4**

**Restraints4**

**Test Models4**

**Project description:**

This project’s goal was to create a website that can save student events at universities. The database was created using Heroku with a ClearDB MySQL extension and the MySQL workbench while the front end and API were implemented utilizing the Javascript language.

Our front end was designed to handle three types of users: Students, Admins, and Super-Admins. The Students can be associated with one or more RSO’s, but are not required to be in one to have an account on the site. Admins are the students who represent one or more RSOs and can create Public, Private, and RSO events. The public events that the Admin creates must be verified however before they display in the feeds of other students. The third type of user is the Super-Admins whose represents a particular university. The Super Admins are responsible for creating RSOs and assigning Admin access to an elected member of the RSO. The Super Admin can create public and private events for the university and approve of the public events that an Admin has created.

Events that are set as public can be seen by everyone. Events that are private can only be seen by users who were invited. RSO events can only be viewed by Students who are members of that RSO. The location of these events can be viewed on the site’s map using latitudinal and longitudinal coordinates. Events are not allowed to overlap and are forced by the API to maintain a sequential ordering.

ER Diagram:

Diagram

Description automatically generated

Front End GUI:  
GUI: Platform, JavaScript

Front end was designed to restrict the creation of an RSO to require at least five members. The RSO cannot be created otherwise.

Graphical user interface, application

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

**Relational Data model:**

create table university

(

numOfStudents INTEGER,

description text,

uniName char(225) not null,

location char(20),

primary key (uniName)

);

create table users

(

UID integer not null unique auto\_increment,

username char(20) not null unique,

password char(20) not null,

userType char(3) not null,

uniName char(20) not null,

#Valid input: SUP, ADM, STU. Constraint needed

constraint checkType check(userType = "SUP" or userType = "ADM" or userType = "STU"),

PRIMARY KEY(UID),

foreign key(uniName) references university(uniName)

);

create table RSO

(

domainEmail char(20),

RID integer not null unique AUTO\_INCREMENT,

UID integer not null,

rsoName char(225) not null,

uniName char(225) not null,

ContactNumber char(12),

ContactEmail char(30),

description text,

isActive boolean default false,

constraint check\_num check (ContactNumber not like '%[^0-9]%'),

PRIMARY KEY (RID),

foreign key(uniName) references university(uniName),

foreign key(UID) references users(UID)

);

create table location

(

LocID integer not null unique auto\_increment,

longitude char(10),

latitude char(10),

locName char(20),

Primary key(locID)

);

create table Eventss

(

Date char(10),

Start time not null,

End time not null,

name char(30) not null,

LocID integer not null,

ContactName char(30),

ContactNumber char(12),

ContactEmail char(30),

Category char(3),

Description text,

RID integer,

UID integer not null,

uniName char(255) not null,

approved boolean default false,

PRIMARY KEY (Date, Start, End),

foreign key(LocID) references location(LocID),

foreign key(RID) references rso(RID),

foreign key(UID) references users(UID),

foreign key(uniName) references university(uniName),

constraint check\_num check (ContactNumber not like '%[^0-9]%')

);

# Disabled this section of code because I do not have the SUPER permission from Heroku to implement it

# The intention was to query the eventss table to see if the insertion would cause an overlap

# We decided to handle this constraint using our API instead as a result

#DELIMITER $$

#create trigger time\_overlap before insert on eventss for each row

# begin

# if exists(select StartT, EndT from eventss e join inserted i on

# ((e.Start < i.End) and (e.End > i.Start))) then

# signal sqlstate '45'; end if; end;$$

create table joined

(

UID integer not null,

RID integer not null,

primary key(UID, RID),

foreign key(UID) references users(UID),

foreign key(RID) references rso(RID)

);

create table comments

(

CID integer not null unique auto\_increment,

textfield text,

rating integer,

UID integer not null,

Date char(10) not null,

Start time not null,

End time not null,

Primary key(CID),

foreign key(UID) references users(UID),

foreign key(Date,Start,End) references eventss(Date,Start,End)

);

Sample Data Inserted Into Table:

# Test uni data

insert into university (uniName, description, numOfStudents, location) VALUES('UCF','a place', 20,'Orlando');

insert into university (uniName, description, numOfStudents, location) VALUES('USF','a place', 20,'Tampa');

insert into university (uniName, description, numOfStudents, location) VALUES('UF','a place', 20,'Gainesville');

# Test student data

insert into users (username,password,userType,uniName) VALUES('StuA','360NoScope', 'STU', 'UCF');

insert into users (username,password,userType,uniName) VALUES('StuB','p455w0rd', 'STU', 'UCF');

insert into users (username,password,userType,uniName) VALUES('StuC','hiMom!', 'STU', 'UCF');

insert into users (username,password,userType,uniName) VALUES('StuD','d4t4', 'STU', 'UCF');

insert into users (username,password,userType,uniName) VALUES('StuE','360NoScope', 'STU', 'USF');

insert into users (username,password,userType,uniName) VALUES('StuF','p455w0rd', 'STU', 'USF');

insert into users (username,password,userType,uniName) VALUES('StuG','hiMom!', 'STU', 'USF');

insert into users (username,password,userType,uniName) VALUES('StuH','d4t4', 'STU', 'USF');

insert into users (username,password,userType,uniName) VALUES('StuI','360Missed', 'STU', 'UF');

insert into users (username,password,userType,uniName) VALUES('StuJ','f4il', 'STU', 'UF');

insert into users (username,password,userType,uniName) VALUES('StuK','ex4mplePass', 'STU', 'UF');

insert into users (username,password,userType,uniName) VALUES('StuL','l4m3P4ssw0rd', 'STU', 'UF');

insert into users (username,password,userType,uniName) VALUES('StuM','itsAPass', 'STU', 'UCF');

insert into users (username,password,userType,uniName) VALUES('StuN','legitPass', 'STU', 'UCF');

insert into users (username,password,userType,uniName) VALUES('StuO','ImRunning', 'STU', 'USF');

insert into users (username,password,userType,uniName) VALUES('StuP','outOf', 'STU', 'USF');

insert into users (username,password,userType,uniName) VALUES('StuQ','jokePass', 'STU', 'UF');

insert into users (username,password,userType,uniName) VALUES('StuR','wordsHere', 'STU', 'UF');

insert into users (username,password,userType,uniName) VALUES('StuS','password1', 'STU', 'UCF');

insert into users (username,password,userType,uniName) VALUES('StuT','passwprd2', 'STU', 'USF');

#Test admin data

insert into users (username,password,userType,uniName) VALUES('AdmA','securePass1', 'ADM', 'UCF');

insert into users (username,password,userType,uniName) VALUES('AdmB','securePass2', 'ADM', 'UCF');

insert into users (username,password,userType,uniName) VALUES('AdmC','securePass3', 'ADM', 'UCF');

#Test super admin data

insert into users (username,password,userType,uniName) VALUES('SupA','AbsSecurity1', 'SUP', 'UCF');

#Test rso,location,event data

insert into location(longitude,latitude,locName) VALUES('-81.2001','28.6024','exLoc');

insert into rso(domainEmail,UID,rsoName,uniName,ContactNumber,ContactEmail,description) values('exampleDom@gmail.com',204,'fratBoys','UCF','813-813-8131','example1@gmail.com','Join ussss');

insert into eventss(Date,Start,End,name,ContactName,ContactNumber,ContactEmail,Category,Description,LocID,RID,UID,uniName) VALUES('1/1/2020','9:30','12:15','testEvent','John Doe', '813-480-4808','goKnights@gmail.com','rso','An example event',4,4,204,'UCF');

insert into location(longitude,latitude,locName) VALUES('-81.200079','28.6024580','Campus');

insert into rso(domainEmail,UID,rsoName,uniName,ContactNumber,ContactEmail,description) values('exampleDom@gmail.com',224,'Theta Beta Pi','UCF','813-813-8131','example2@gmail.com','One of us...One of us...');

insert into eventss(Date,Start,End,name,ContactName,ContactNumber,ContactEmail,Category,Description,LocID,RID,UID,uniName) VALUES('1/2/2020','12:30','13:30','PlanningEvent','John Doe', '813-480-4808','goKnights@gmail.com','rso','An example event',4,4,224,'UCF');

insert into location(longitude,latitude,locName) VALUES('-81.200079','28.6024580','awesomeLoc');

insert into rso(domainEmail,UID,rsoName,uniName,ContactNumber,ContactEmail,description) values('exampleDom@gmail.com',224,'Tau Zeta Pi','UCF','813-813-8131','example3@gmail.com','The work is unending!');

insert into eventss(Date,Start,End,name,ContactName,ContactNumber,ContactEmail,Category,Description,LocID,RID,UID,uniName) VALUES('2/4/2020','12:30','13:30','LFEvent','John Doe', '813-480-4808','goKnights@gmail.com','pri','An example event',4,null,224,'UCF');

insert into location(longitude,latitude,locName) VALUES('-81.200079','28.6024580','dumbName');

insert into rso(domainEmail,UID,rsoName,uniName,ContactNumber,ContactEmail,description) values('exampleDom@gmail.com',214,'Zeta Zeta Alpha','UCF','813-813-8131','example4@gmail.com','Dont join us...');

insert into eventss(Date,Start,End,name,ContactName,ContactNumber,ContactEmail,Category,Description,LocID,RID,UID,uniName) VALUES('8/12/2020','1:30','15:30','RFEvent','John Doe', '813-480-4808','goKnights@gmail.com','pub','An example event',4,null,224,'UCF');

#Test joined data

insert into joined(UID,RID) VALUES(204,4);

insert into joined(UID,RID) VALUES(224,14);

insert into joined(UID,RID) VALUES(224,24);

insert into joined(UID,RID) VALUES(224,34);

insert into joined(UID,RID) values(4,4);

insert into joined(UID,RID) values(14,4);

insert into joined(UID,RID) values(24,4);

insert into joined(UID,RID) values(34,4);

insert into joined(UID,RID) values(44,14);

insert into joined(UID,RID) values(54,14);

insert into joined(UID,RID) values(64,14);

insert into joined(UID,RID) values(74,14);

insert into joined(UID,RID) values(84,24);

insert into joined(UID,RID) values(94,24);

insert into joined(UID,RID) values(104,24);

insert into joined(UID,RID) values(114,24);

insert into joined(UID,RID) values(124,34);

insert into joined(UID,RID) values(134,34);

insert into joined(UID,RID) values(144,34);

insert into joined(UID,RID) values(154,34);

#Test comments data

insert into comments(textfield,rating,UID,Date,Start,End) values("Looks awesome!",4,4,'8/12/2020','1:30','15:30');

insert into comments(textfield,rating,UID,Date,Start,End) values("I don't know bro. Looks kinda sus...",1,14,'8/12/2020','1:30','15:30');

insert into comments(textfield,rating,UID,Date,Start,End) values("This is gonna be good",3,24,'8/12/2020','1:30','15:30');

insert into comments(textfield,rating,UID,Date,Start,End) values("I hate this",1,24,'8/12/2020','1:30','15:30');

insert into comments(textfield,rating,UID,Date,Start,End) values("There is no god and this event is proof",1,24,'8/12/2020','1:30','15:30');

SQL Examples:

``Several SQL queries to display events—public, private, and RSO-- (part of the processing of the ‘View Event’ request by a user with a specific role), show results o SQL statements of interest (optional), e.g., advanced SQL queries``

**Constraint Enforcement:**

Event to be held at same location with overlapping times with existing event:Graphical user interface, application

Description automatically generated

Show SCREENSHOTS of error messages/warnings when the following events/attempts are made:

An admin who is not the Admin of the RSO attempts to create an event for that RSO: Show an error message.