Concordia University
Dept. of Computer Science & Software Engineering
Fall 2019

COMP 353/2 F Databases

To:
Dr. Bipin C. Desai

Charles Abou Haidar 40024373
Jessica Allaire 40015912
Mario Bastiampillai 40016804

December 1st 2019

TABLE OF CONTENTS

Project Description

The requirements specified that a Share, Contribute and Comment system be implemented using MySQL. A web based interface was to be developed and all tables utilized should be normalized to the 3rd normal form. The entire system should facilitate the organization and scheduling of various events held by different factions. Events should have participants who may form groups, in which they are free to communicate and collaborate to the success of their event through the use of messages and postings. In terms of implementation, several components representing various roles will be developed, with the overall interactions between these components forming the system.

Furthermore, specific functionalities were required to ensure the robustness of the system. These include privacy and security handling, financial management as well as statistical reporting.

Assumptions

TBD has quite a few assumptions. To make the assumptions as clear as possible, the assumptions will be listed according to the roles, the objects, and then more vaguely to the system.

The roles for this system are System Administrator (SA), Controller (C), Event Manager (EM), Event Administrator (EA), Participant (P), and Visitor.

System Administrator

- There is only one SA for the whole site.
- The SA can do all actions admissible by the other roles.
- The SA can be a participant in an event.
- The SA can be an EM for an event.
- The SA can be an EA for an event.
- The aim of the SA is to keep the system alive and running well so that users can perform a maximum amount of useful work.
- The SA account is only accessible by one person.
- The SA account is secure and cannot be hacked.
- The SA is smart and trustworthy with the information of the system.

Controller

- The controller is one person for the whole site.
- The Controller can be a participant in an event.
- The Controller can be an EM for an event.
- The Controller can be an EA for an event.
- The aim of the Controller is to control the basic system attributes.

Event Manager

- The EM is a role that is responsible for creating an event.
- The EM pays the system to host their event on the website.
- The EM chooses the design from a pre-given set of options.
- The EM can extend the life by charging an extra amount to their cost.
- The EM can only see the events they are managing and the events they are participating in.

Event Administrator

- The EA is a role that deals with the event online.
- The EA is responsible for adding the initial participant to an event.
- There is only one EA per event.
- An EA can be a participant in another event.

- The EA is responsible for posting content onto the events it is administering.

Participant

- A participant is one that has been added or invited to join an existing event.
- The participant does not need to pay for anything.
- A participant can only see their events and groups.
- A participant can create any number of groups per event they are participating.
- A participant is the one who gives people permission to edit certain posts.

Event

- Events have one EA and one EM.
- Events do not show all the participants.
- Event pages show only the posts and the descriptions.
- Events can have

Group

- Groups are created by participants inside of an event.
- A group can have an event associated with it.

Messages

- Messages come in three forms.
- Messages can be notifications, messages from others in a group, and an invitation to/from an event/group.
- Once a message is read, it is not shown unless forced.
- Once a message is deleted, it may never be retrieved.

System

- The cost to host an event is a fixed price per day.
- Extra charge for more bandwidth or storage is a fixed rate.
- Bandwidth is a set amount for each event unless changed by the controller.
- Storage is a set amount for each event unless changed by the controller.
- Visitors can only see the front page and the login page.
- An event has a duration and a period. The duration is the amount of time the event lasts. The period is how long the event is hosted on the website for.
- The costs for all this is not for the event but rather for hosting it on the system.

Limitations

Throughout the development of the project, various limitations were realized both in terms of design and implementation. Design limitations were constraints that were either specified within the project requirements or were details of our own choosing, while implementation limitations were constraints or restrictions due to the way the software was built.

Design limitations

- It was decided that only participants within a group may be able to communicate with other participants within the same group. Participants outside of a group may not be able to communicate with others.
- Participants can only join a group through a received invitation. Participants outside of a group may not join by themselves.

Implementation limitations

- The email messaging system was implemented in an autonomous fashion, emails are generated randomly by participants and notifications will appear for the recipients.

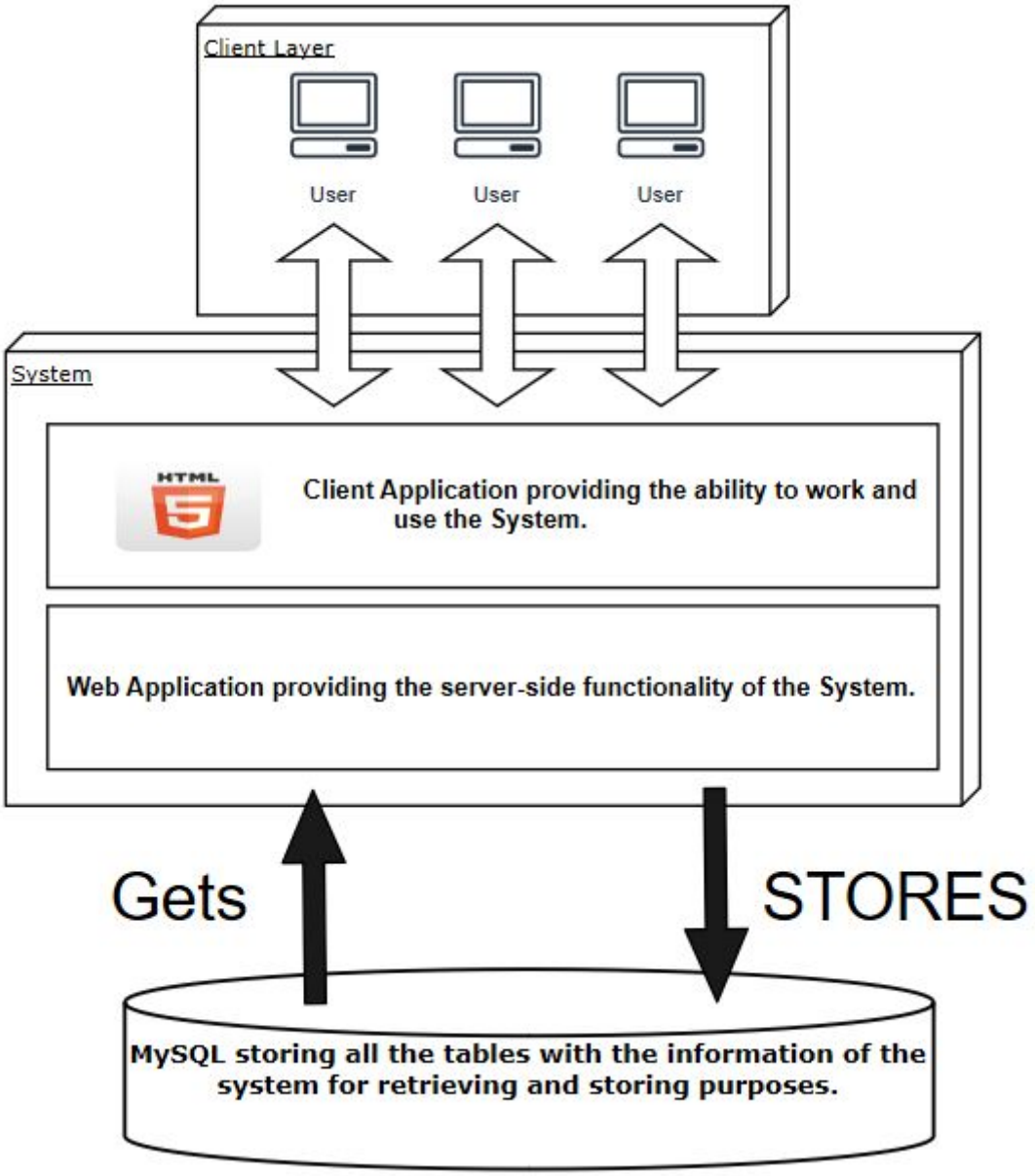
Applications Supported

The system is able to run on a variety of browser applications including Microsoft Edge, Google Chrome, Mozilla Firefox etc. This ensures the adaptability of the system, meaning a large demographic is able to use the system. The system is also able to run on numerous phone web browsers as well. This means that the web design is responsive across multiple different platforms.

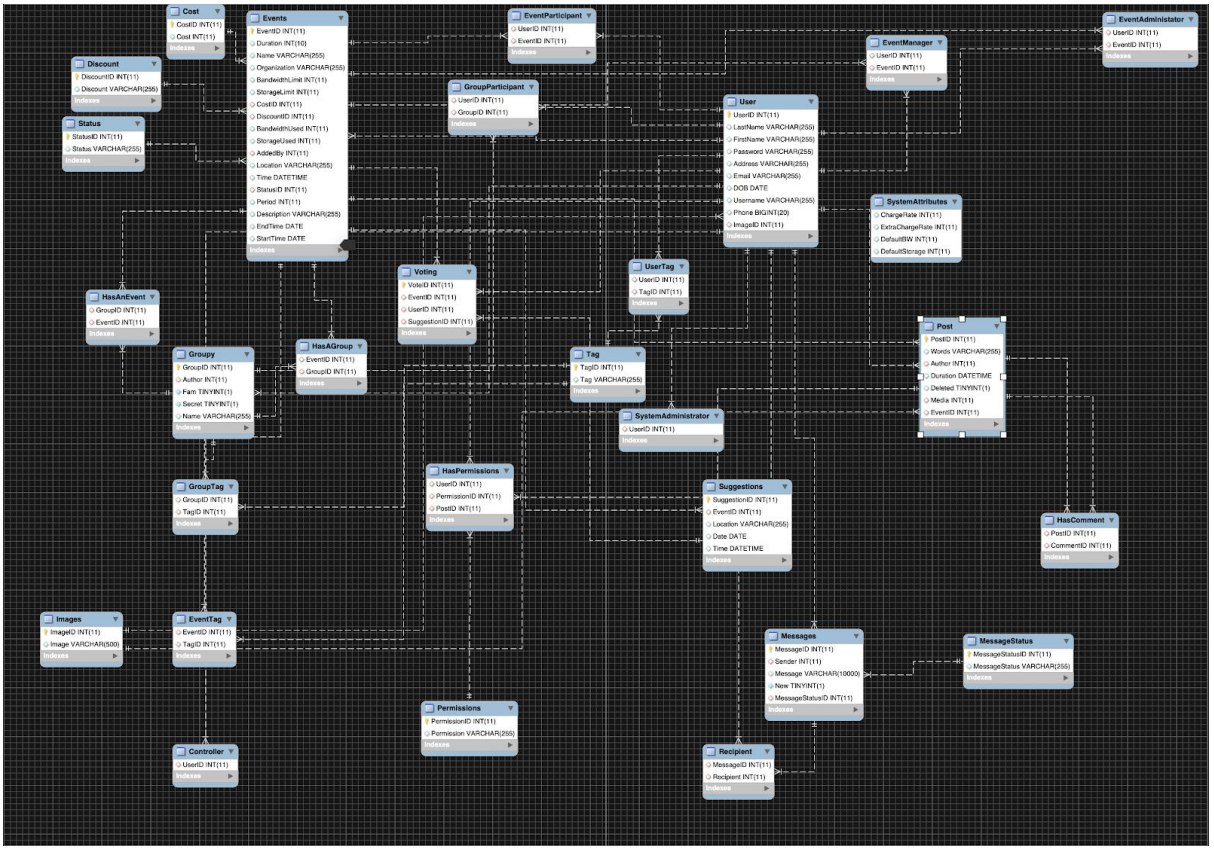
Aside from this, the system will also make use of online pay systems such as Paypal and Moneris, which will be integrated into the software. These online pay systems will be used for financing management.

Architectural Design

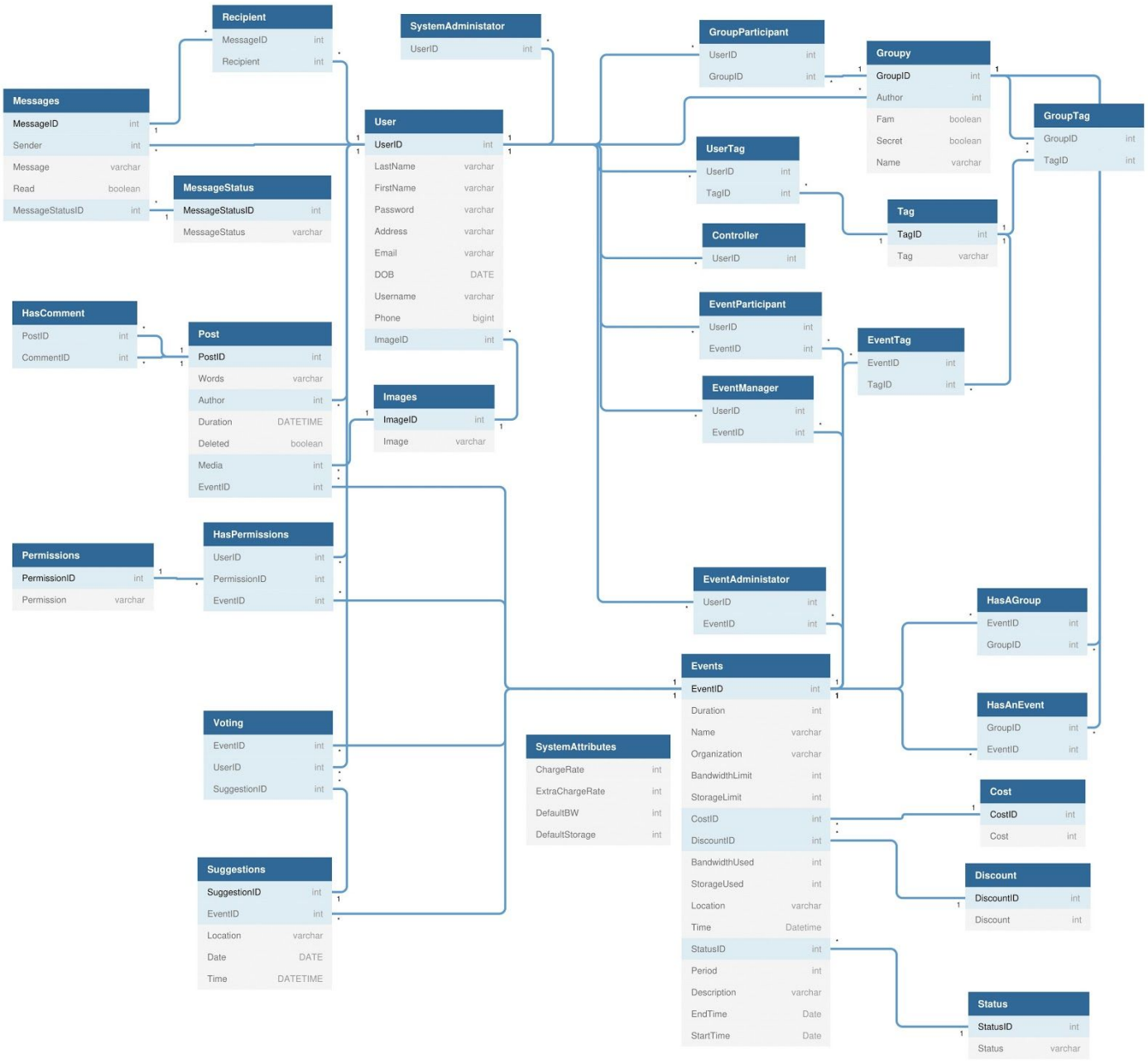
Layered (n-tier) architecture is the architecture used for this system. This is one of the most basic architectures and works by the users' actions making their way down to the database by traveling through all the different layers. These layers include the frontend which is the html and css, then making its way into the backend of PHP and JavaScript, that goes into the MySQL database (the last layer).



E-R Diagrams


Cost
CostID INT(11)
Cost INT(11)
Events
EventID INT(11)
Duration INT(10)
Name VARCHAR(255)
Organization VARCHAR(255)
BandwidthLimit INT(11)
StorageLimit INT(11)
CostID INT(11)
DiscountID INT(11)
BandwidthUsed INT(11)
StorageUsed INT(11)
AddedBy INT(11)
Location VARCHAR(255)
Time DATETIME
StatusID INT(11)
Period INT(11)
Description VARCHAR(255)
EndTime DATE
StartTime DATE
EventParticipant
UserID INT(11)
EventID INT(11)
EventManager
UserID INT(11)
EventID INT(11)
EventAdministator
UserID INT(11)
EventID INT(11)
Discount
DiscountID INT(11)
Discount VARCHAR(255)
GroupParticipant
UserID INT(11)
GroupID INT(11)
User
UserID INT(11)
LastName VARCHAR(255)
FirstName VARCHAR(255)
Password VARCHAR(255)
Address VARCHAR(255)
Email VARCHAR(255)
DOB DATE
Username VARCHAR(255)
Phone BIGINT(20)
ImageID INT(11)
Status
StatusID INT(11)
Status VARCHAR(255)
SystemAttributes
ChargeRate INT(11)
ExtraChargeRate INT(11)
DefaultBW INT(11)
DefaultStorage INT(11)
Voting
VoteID INT(11)
EventID INT(11)
UserID INT(11)
SuggestionID INT(11)
UserTag
UserID INT(11)
TagID INT(11)
HasAnEvent
GroupID INT(11)
EventID INT(11)
Post
PostID INT(11)
Words VARCHAR(255)
Author INT(11)
Duration DATETIME
Deleted TINYINT(1)
Media INT(11)
EventID INT(11)
Groupy
GroupID INT(11)
Author INT(11)
Fam TINYINT(1)
Secret TINYINT(1)
Name VARCHAR(255)
HasAGroup
EventID INT(11)
GroupID INT(11)
Tag
TagID INT(11)
Tag VARCHAR(255)
SystemAdministrator
UserID INT(11)
GroupTag
GroupID INT(11)
TagID INT(11)
HasPermissions
UserID INT(11)
PermissionID INT(11)
PostID INT(11)
Suggestions
SuggestionID INT(11)
EventID INT(11)
Location VARCHAR(255)
Date DATE
Time DATETIME
HasComment
PostID INT(11)
CommentID INT(11)
Images
ImageID INT(11)
Image VARCHAR(500)
EventTag
EventID INT(11)
TagID INT(11)
Messages
MessageID INT(11)
Sender INT(11)
Message VARCHAR(10000)
New TINYINT(1)
MessageStatusID INT(11)
MessageStatus
MessageStatusID INT(11)
MessageStatus VARCHAR(255)
Permissions
PermissionID INT(11)
Permission VARCHAR(255)
Controller
UserID INT(11)
Recipient
MessageID INT(11)
RecipientINT(11)

Relational Database Design



3NF Solution

For our system to respect the 3rd normal form, it must satisfy the 2nd and 1st normal forms as well. The criteria for the 1st normal form requires that every row must be unique as well as every cell containing a single value. The criteria for the 2nd normal form requires that there must be a single column primary key and must also satisfy the 1st normal form requirements. The criteria for the 3rd normal form demands that no transitive dependencies be present and must also conform with the 2nd normal form requirements.

To ensure that 1st normal form standards were being followed each row was made sure to be unique and arrays were not used since that would violate the single value per cell requirement. The following diagram depicts this criteria:

Figure 1:

Event									
Event_ID	Duration	name	organization	bandwidth_Limit	storage_limit	Cost_ID (FK)	Discount_ID (FK)	Bandwidth_used	Storage_used
Location	Time	Date	status_ID (FK)						

Figure 1 is representative of all other tables following the same 1st normal form standard.

To ensure that the 2nd normal form standards were being followed, our solution made use of many separate and modular tables, as well as joint tables when a relation was needed to be expressed. The use of these joint tables eliminated the problem of having two primary keys within a single table, since both of these primary keys would be now expressed as foreign keys. The following diagram depicts this criteria:

Figure 2:

User									
ID (PK)	First Name	Last Name	Password	Address	Email	DOB	Username	Phone Number	ImageID (FK)

Event									
Event_ID	Duration	name	organization	bandwidth_Limit	storage_limit	Cost_ID (FK)	Discount_ID (FK)	Bandwidth_used	Storage_used
Location	Time	Date	status_ID (FK)						

Event_Participant		Group_Participant	
User_ID (FK)	Event_ID (FK)	User_ID (FK)	Group_ID (FK)

Figure 2 is representative of all other tables following the same 2nd normal form standard.

To ensure that the 3rd normal form standards were being followed, redundant relationships between attributes were excluded within tables. The following diagram depicts this criteria:

Figure 3:

Event									
Event_ID	Duration	name	organization	bandwidth_Limit	storage_limit	Cost_ID (FK)	Discount_ID (FK)	Bandwidth_used	Storage_used
Location	Time	Date	status_ID (FK)						

Figure 3 is representative of all other tables following the same 3rd normal form standard.

Member Responsibility

- Jessica Allaire: group leader, was in charge of assigning the tasks
  Worked on the front-end and back-end of the following pages:
    - Website homepage
    - Event Admin
    - Event Manager
    - Group
    - Footer and header

  Mario Bastiampillai
  Worked on the front-end and back-end of the following pages:
    - System Admin
    - Controller
    - Event

- Charles Abou Haidar
  Worked on the front-end and back-end of the following pages:
    - User homepage
    - Settings(User Settings)
    - Profile

  The following was worked on by the entire team together during meetings:

- Database design:
    - Making sure that the database is in 3NF
    - Attributes
    - Assignment of primary and foreign keys
    - Mock tables for the entire database
- SQL Queries

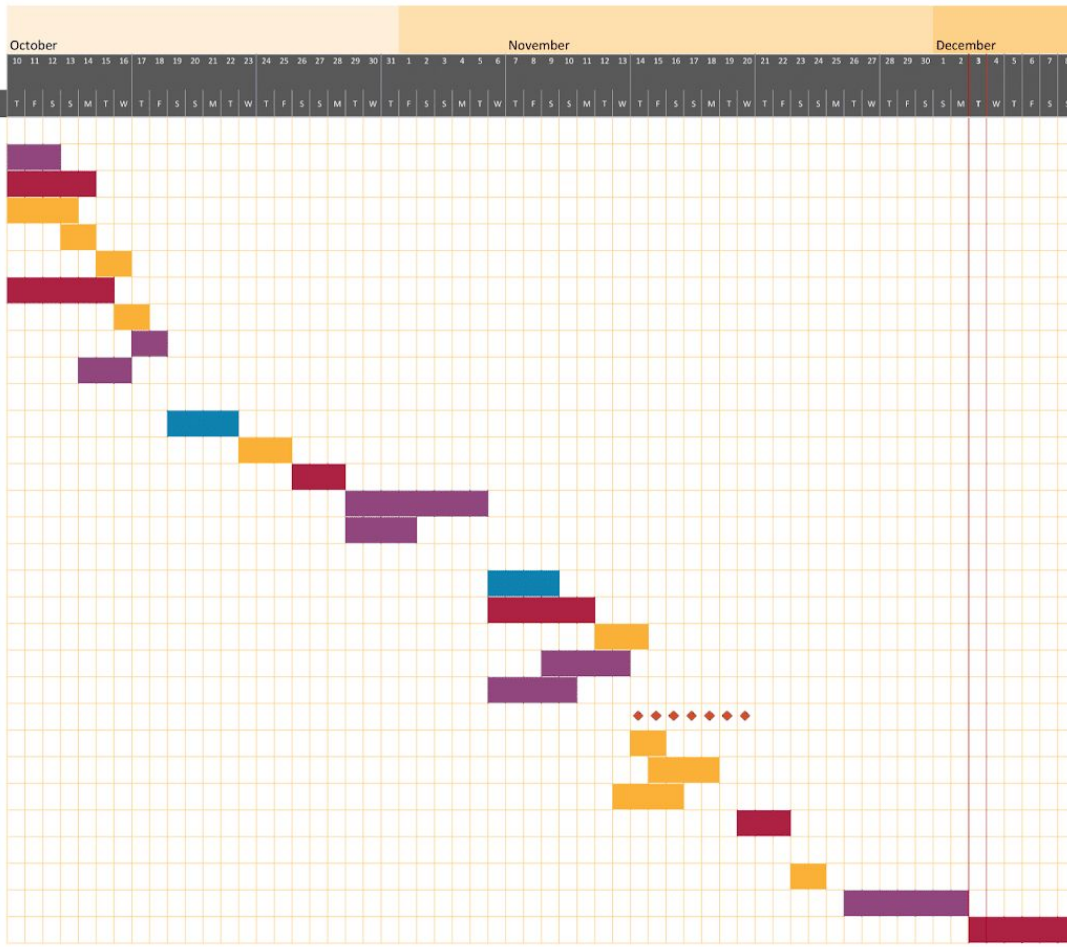
Gantt Chart

TBD (Share, Comment, Contribute)
TBD Inc.
Mario Bastiampillai 40016804
Charles Abou Haidar 40024373
Jessica Allaire 40015912

Project Start Date: 2019-10-10

Legend: On Track | Low Risk | Med Risk | High Risk | Unassigned

Milestone Desc.	.	Assigned To	Progress	Start	No. Days
Front End					
System	Med Risk	Jess	85%	2019-10-10	3
SA	High Risk	Mario	60%	2019-10-10	5
Ctrler	Low Risk	Mario	100%	2019-10-10	4
EA	Low Risk	Jess	100%	2019-10-13	2
EM	Low Risk	Jess	90%	2019-10-15	2
User	High Risk	Charles	100%	2019-10-10	6
Visitor	Low Risk	Charles	100%	2019-10-16	2
Event	Med Risk	Mario	80%	2019-10-17	2
Group	Med Risk	Jess	70%	2019-10-14	3
Database					
ERD	On Track	Charles	100%	2019-10-19	4
RD	Low Risk	Jess	100%	2019-10-23	3
3NF Sol.	High Risk	Mario	100%	2019-10-26	3
Mock DB	Med Risk	All	80	2019-10-29	8
SQL	Med Risk	M & C	100%	2019-10-29	4
Back End					
System	On Track	Charles	60%	2019-11-06	4
SA	High Risk	Jess	60%	2019-11-06	6
Ctrler	Low Risk	Jess	40%	2019-11-12	3
EA	Med Risk	Charles	90%	2019-11-09	5
EM	Med Risk	Mario	60%	2019-11-06	5
User	Goal	Mario	80%	2019-11-14	7
Visitor	Low Risk	Charles	100%	2019-11-14	2
Event	Low Risk	Jess	60%	2019-11-15	4
Group	Low Risk	Mario	20%	2019-11-13	4
DB	High Risk	All	100%	2019-11-20	3
Report					
Outline	Low Risk	All	100%	2019-11-23	2
Draft	Med Risk	All	100%	2019-11-26	7
Final	High Risk	All	50%	2019-12-03	7



Detailed Analysis of Coding the Website

The website was coded primarily with the use of PHP, CSS, JavacScript and SQL queries. PHP pages were used to construct the front end of the site since it included HTML tags as well as CSS links. CSS was used for aesthetic purposes of the front end, while JavaScript was used to implement specific functions related to tasks the different roles had to do. SQL queries were used within the PHP pages to perform actions related to the manipulation of the database.

System Admin

The System Admin PHP page made use of HTML tags to implement the CSS link, header, buttons as well as the forms needed to obtain user input. These buttons were for “Add Event”, “Assign Manager”, “Add participant”, “Change Event Status”, “View List of Events” and “View List of Participants” functionalities. A form was implemented for every single one of these tasks which would appear when the corresponding button was clicked. A separate CSS file was used to hold everything related to the style of the page itself, buttons and forms. A separate JavaScript file was used to hold everything related to performing the functions related to opening the forms once the buttons were clicked.

Controller

The Controller PHP page made use of HTML tags to implement the CSS link, header, buttons as well as the forms needed to obtain user input. These buttons were for “Set charge rate”, “Set bandwidth”, “Set discount”, “Set storage” and “Set period” functionalities. A form was implemented for every single of of these tasks which would appear when the corresponding button was clicked. A separate CSS file was used to hold everything related to the style of the page itself, buttons and forms. A separate JavaScript file was used to hold everything related to performing the functions related to opening the forms once the buttons were clicked.

Event

The Event PHP page made use of HTML tags to implement the CSS link, header, buttons as well as the forms needed to obtain the user input. These buttons were for “View Event” and “Post Event” functionalities. A form was implemented for each of these tasks which would appear when the corresponding button was clicked. A separate CSS file was used to hold everything related to the style of the page itself, buttons and forms. A separate JavaScript file was used to hold everything related to performing the functions related to opening the forms once the buttons were clicked.

Interface Design Rational

Queries

Creation of tables:

```
CREATE TABLE User (
  UserID int NOT NULL AUTO_INCREMENT PRIMARY KEY,
  LastName varchar(255),
  FirstName varchar(255),
  Password varchar(255),
  Address varchar(255),
  Email varchar(255),
  DOB DATE,
  Username varchar(255),
  Phone bigint,
  ImageID int,
  FOREIGN KEY (ImageID) REFERENCES Images(ImageID),
);
CREATE TABLE Events (
  EventID int NOT NULL AUTO_INCREMENT PRIMARY KEY,
  Duration int(10),
  Name varchar(255),
  Organization varchar(255),
  BandwidthLimit int,
  StorageLimit int,
  CostID int,
  FOREIGN KEY (CostID) REFERENCES Cost(CostID),
  DiscountID int,
  FOREIGN KEY (DiscountID) REFERENCES Discount(DiscountID),
  BandwidthUsed int,
  StorageUsed int,
  AddedBy int,
  FOREIGN KEY (AddedBy) REFERENCES User(UserID),
  Location varchar(255),
  Time Datetime,
  Date Date,
  StatusID int,
  FOREIGN KEY (StatusID) REFERENCES Status(StatusID)
);
CREATE TABLE SystemAdministrator (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID)
);

CREATE TABLE Post (
  PostID int NOT NULL AUTO_INCREMENT PRIMARY KEY,
  Words varchar(255),
```

```
  Author int,
  FOREIGN KEY (Author) REFERENCES User(UserID),
  Duration DATETIME,
  Deleted boolean,
  Media int
);
CREATE TABLE Groupy (
  GroupID int AUTO_INCREMENT PRIMARY KEY,
  Author int,
  FOREIGN KEY (Author) REFERENCES User(UserID),
  Fam boolean NOT NULL,
  Secret boolean NOT NULL,
  Name varchar(255)
);
CREATE TABLE Controller (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID)
  );
CREATE TABLE EventManager (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID)
);
CREATE TABLE EventAdministrator (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID)
);
CREATE TABLE EventParticipant (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID)
);
CREATE TABLE GroupParticipant (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
  GroupID int,
  FOREIGN KEY (GroupID) REFERENCES Groupy(GroupID)
);
CREATE TABLE SystemAttributes (
  ChargeRate int,
  ExtraChargeRate int,
  DefaultBW int,
```

```
  DefaultStorage int
 );
CREATE TABLE Status (
  StatusID int AUTO_INCREMENT PRIMARY KEY,
  Status varchar(255)
);
CREATE TABLE Discount (
  DiscountID int AUTO_INCREMENT PRIMARY KEY,
  Discount int
);
CREATE TABLE Cost (
  CostID int AUTO_INCREMENT PRIMARY KEY,
  Cost int
);
CREATE TABLE HasAGroup (
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID),
  GroupID int,
  FOREIGN KEY (GroupID) REFERENCES Groupy(GroupID)
);
CREATE TABLE HasAnEvent (
  GroupID int,
  FOREIGN KEY (GroupID) REFERENCES Groupy(GroupID),
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID)
);
CREATE TABLE Voting (
  VoteID int AUTO_INCREMENT PRIMARY KEY,
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID),
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
  SuggestionID int,
  FOREIGN KEY (SuggestionID) REFERENCES Suggestions(SuggestionID)
);
CREATE TABLE Suggestions (
  SuggestionID int AUTO_INCREMENT PRIMARY KEY,
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID),
  Location varchar(255),
  Date DATE,
  Time DATETIME
);
CREATE TABLE Messages (
  MessageID int AUTO_INCREMENT PRIMARY KEY,
  Sender int,
```

```
  FOREIGN KEY (Sender) REFERENCES User(UserID),
  Message varchar(255),
  New boolean not null,
  MessageStatusID int,
  FOREIGN KEY (MessageStatusID) REFERENCES MessageStatus(MessageStatusID)
);
CREATE TABLE MessageStatus (
  MessageStatusID int not null auto_increment primary key,
  MessageStatus varchar(255)
);
CREATE TABLE Recipient (
MessageID int,
FOREIGN KEY (MessageID) REFERENCES Messages(MessageID),
Recipient int,
FOREIGN KEY (UserID) REFERENCES User(UserID)
);
CREATE TABLE EventTag (
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID),
  TagID int,
  FOREIGN KEY (TagID) REFERENCES Tag(TagID)
);
CREATE TABLE UserTag (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
  TagID int,
  FOREIGN KEY (TagID) REFERENCES Tag(TagID)
);
CREATE TABLE GroupTag (
  GroupID int,
  FOREIGN KEY (GroupID) REFERENCES Groupy(GroupID),
  TagID int,
  FOREIGN KEY (TagID) REFERENCES Tag(TagID)
);
CREATE TABLE Tag (
  TagID int AUTO_INCREMENT PRIMARY KEY,
  Tag varchar(255)
);
CREATE TABLE Permissions (
  PermissionID int AUTO_INCREMENT PRIMARY KEY,
  Permission varchar(255)
);
CREATE TABLE HasPermissions (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
  PermissionID int,
```

```
  FOREIGN KEY (PermissionID) REFERENCES Permissions(PermissionID),
  PostID int,
  FOREIGN KEY (PostID) REFERENCES Post(PostID)
);
CREATE TABLE HasComment (
   PostID int,
   FOREIGN KEY (PostID) REFERENCES Post(PostID),
   CommentID int,
   FOREIGN KEY (CommentID) REFERENCES Post(PostID)
);
CREATE TABLE Images (
   ImageID int AUTO_INCREMENT PRIMARY KEY,
   Image varchar(500)
);
```

Others:

- System Admin
  - Add event:

    Insert into Events (Duration, Name, Organization, Location, Time, Date);

  - Assign Manager

    SELECT EventID, Name, Organization FROM Events;

    SELECT User.UserID, Images.Image, User.FirstName, User.LastName FROM User INNER JOIN Images WHERE User.ImageID = Images.ImageID;

    Insert into EventManager(UserID, EventID);

  - Add Participant

    SELECT EventID, Name, Organization FROM Events;

    SELECT User.UserID, Images.Image, User.FirstName, User.LastName FROM User INNER JOIN Images WHERE User.ImageID = Images.ImageID;

    Insert into EventParticipant(UserID, EventID);

  - Add Group Participant

    SELECT GroupID, Author FROM Groupy;

    SELECT User.UserID, Images.Image, User.FirstName, User.LastName FROM User INNER JOIN Images WHERE User.ImageID = Images.ImageID;

    Insert into GroupParticipant(UserID, GroupID);

  - Change Event Status

    SELECT EventID, Name, StatusID, Status FROM Events INNER JOIN Status WHERE Events.StatusID=Status.StatusID;

UPDATE Events
SET StatusID = StatusValue/ //status picked by SysAdmin
WHERE EventID = EventValue;

- View List Of Events
  SELECT EventID, Name, Organization FROM Events

- View list of EventParticipants
  SELECT UserID FROM EventParticipant WHERE EventParticipant.EventID= EventValue;

- View list of Groups
  SELECT * FROM Events WHERE Events.EventID = HasAGroup.EventID INNER JOIN Groupy WHERE HasAGroup.GroupID = Groupy.GroupID ;

- View list of participants of a Group
  SELECT userID, LastName, FirstName FROM User INNER JOIN GroupParticipant WHERE User.UserID = GroupParticipant.UserID;

- Set charge rate of an Event
  UPDATE SystemAttributes
  SET ChargeRate = chargeRateValue;

- Set Bandwidth limit
  UPDATE Events
  SET BandwidthLimit = bandwidthLimitValue
  WHERE EventID = EventValue;

- Set discount
  SELECT DiscountID, EventID, Name, Organization FROM Events
  INNER JOIN Discount
  WHERE Events.DiscountID = Discount.DiscountId
  AND Events.EventID = EventValue
  UPDATE Discount
  SET Discount = DiscountValue
  WHERE Events.DiscountID = Discount.DiscountId;

- Set Storage
  UPDATE Events
  SET StorageLimit = StorageLimitValue
  WHERE EventID = EventValue;

- Set period
  UPDATE Events

```
SET Period = periodValue
WHERE EventID = EventValue;
```

- Post Content on Event

```
INSERT INTO hasAComment (postID1, postID2)
VALUES(postValue1, postValue2);
```

- Modify Event

```
UPDATE  Events
SET Name=nameValue, Description=descriptionValue
WHERE EventAdministrator.EventID=Events.EventID;
```

- Remove/Add/Update Participant

```
UPDATE EventParticipant
SET UserID = UserIDValue
WHERE EventID = EventValue;
```

- List of archived events

```
SELECT EventID, Name, Organization FROM Events WHERE
Events.EventID = EventIDValue AND Period > 7;
```

- Register Event:

```
INSERT INTO
Events(DurationValue,EventNameValue,OrganizationValue,AuthorValue,Loca
tionValue,TimeValue,Datevalue,DescriptionValue)
```

- Extend Event Life

```
UPDATE Events
SET Period=periodValue
WHERE EventID = EventValue;
```

- Create Group

```
INSERT INTO Groupy(Author, Fam, Secret,
Name)VALUES(AuthorValue, FamValue, SecretValue, NameValue);
```

User Manual

Step 1: Go to:
https://jrc353.encs.concordia.ca/dbProject/System/Homepage.php

Step 2: Click on the Sign In/Sign up button to login or sign up

Step 3: Login with proper credentials or create an account(will automatically be created as a normal user with no special privileges)

After logging in, you will be redirected to the User's home page, where you will be able to use the system according to the type of credentials you are using.

General Information

The system is comprised of many individual components or roles, all working together to form the system as a whole. These components include the System Administrator, Controller, Event Manager, Event Administrator, Participants, etc. all of which are considered to be users of the system. Each of these roles can perform specific functions related to their roles regarding the creation of events and groups, as well as all the intricacies related to managing and manipulating these details.

The System Administrator is capable of performing all important tasks, while the Controller is tasked with setting different properties such as the charge rate, bandwidth etc. The Event Manager is responsible for creating the event while the Event Admin takes care of everything related to the online aspects of that event. Participants are those who have been asked to participate within an event.

Events can have particular groups and members within a group can message each other as well.

Authorized Use Permissions

The idea of role-based access control (RBAC) is the idea that security is based on the roles people have within a system. In this case the following roles are

System Administrator

The SA has the ability to be able to access anything in the System. This means that all the roles that are in this list have only part of the roles the SA have. To see all the access control the SA has, please refer to the rest of the list to avoid repetition.

Controller

The Controller has access to changing the bandwidth and storage of the events. The Controller also has control over the charge rate, the discounts for the events, and the ability to set the periods.

Event Manager

The EM has access to registering new events, event page setups, removing and adding participants, changing the event statuses, extending the life of an event with extra charge, see all their events and their archived events.

Event Administrator

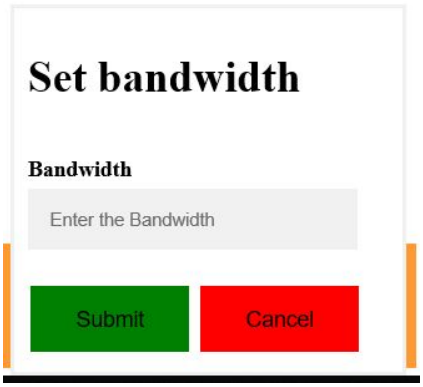
EAs can modify their events but not delete them. They can add, remove and modify participants like the EM. Like the EM, the admin has access to all the participants information. They also have access to posting content into multiple events at once. Finally, they can upload a bunch of participants with a CSV file.

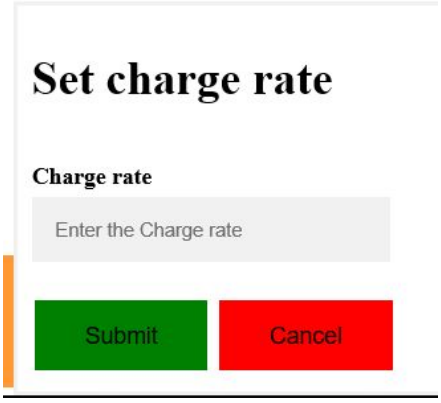
Participant

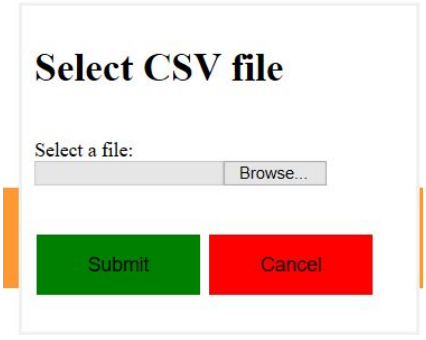
- A participant has access to their events and groups. The have access to messaging those inside the same groups. They have access to their profiles as well as their settings. They have access to posting content and changing the permissions of others for their own content.

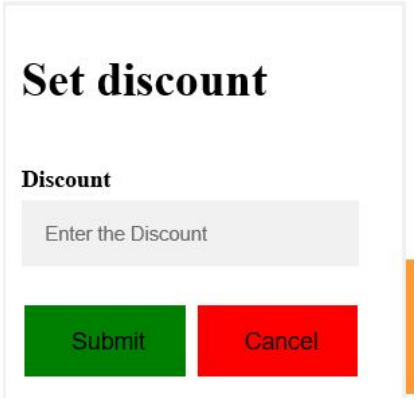
System Summary

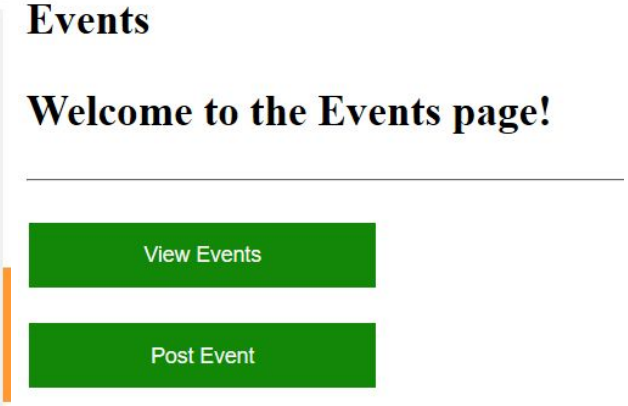
Overview of the User Interface

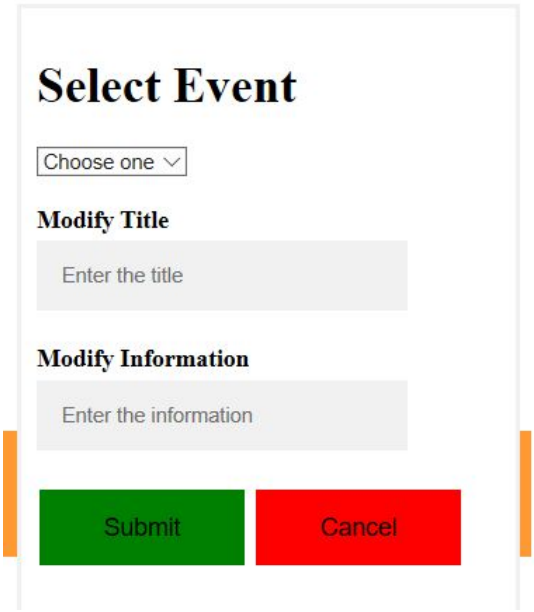


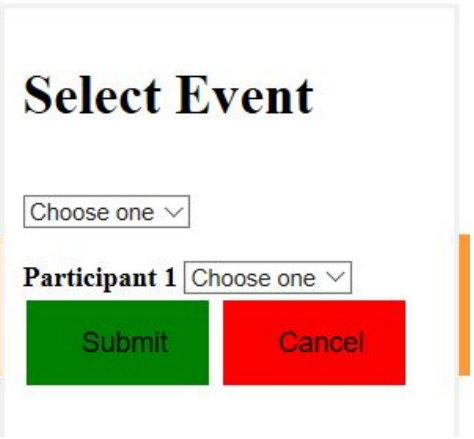


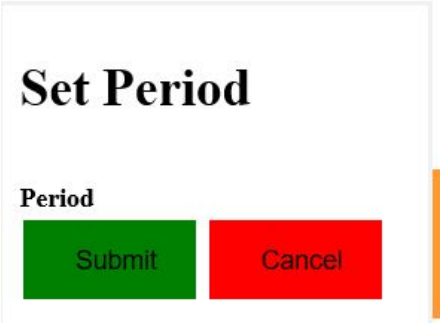




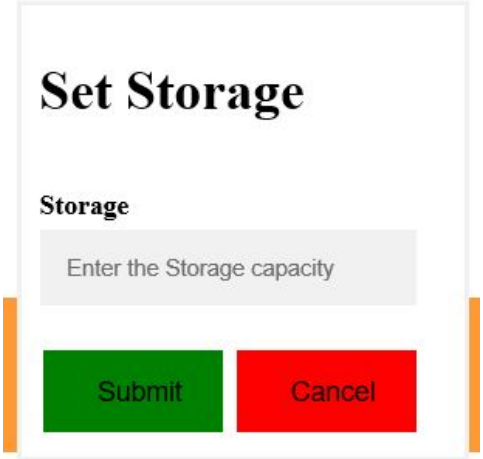



Select Event
Choose one
Modify Title
Enter the title
Modify Information
Enter the information
Submit
Cancel


Select Event
Choose one
Participant 1
Choose one
Submit
Cancel


Set Period
Period
Submit
Cancel


Post Content
Choose one
Post
Enter Content
Submit
Cancel


Set Storage
Storage
Enter the Storage capacity
Submit
Cancel

Homepage

TBD
Share, Contribute, and Comment

Sign Up/Sign In

Welcome to TBD where all your event needs are met!

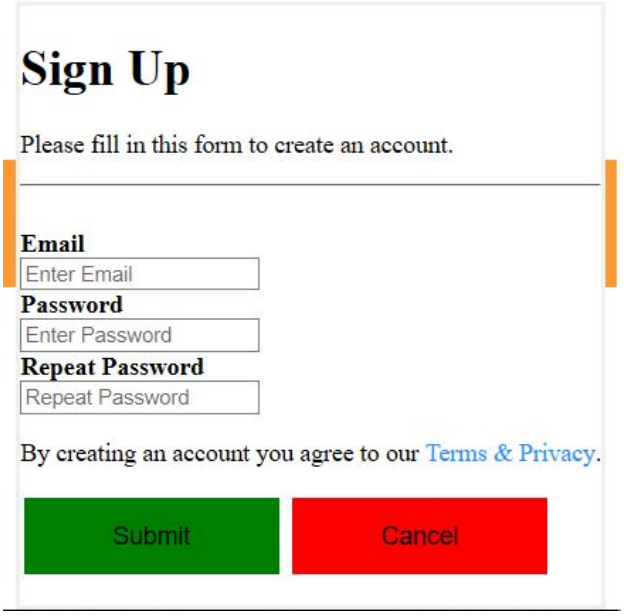
http://localhost/dbProject/System/Homepage.php

TBD is a website devoted to events.
Contact us at support@TBD.ca

TBD
Share, Contribute, and Comment

Visitors


TBD
Share, Contribute, and Comment
Sign Up/Sign In
Sign In
User name or Email
Password
Log in
Sign Up
Controller
Event
Event Admin
Event Manager
Group
System Admin
User
Profile
TBD is a website devoted to events.
Contact us at support@TBD.ca
TBD


Sign Up
Please fill in this form to create an account.
Email
Enter Email
Password
Enter Password
Repeat Password
Repeat Password
By creating an account you agree to our Terms & Privacy.
Submit
Cancel

Members

Administrator

Welcome to your personal homepage! Click on the options below to perform an action!

Register Event

Event Page Setup

Remove/Add/Update Participants

Change Event Status

Extend Event Life (With Charge)

View Events

Welcome to your personal homepage! Click on the options below to perform an action!

Modify Event

Remove/Add/Update Participants

Post Content

Upload CSV

List of Events

List of Participants

List of Archived Events

Git Log

commit b4f200dd1dde1255f5049e6cac4767fac0da4d62 (HEAD -> Database_Mock, origin/Database_Mock)
Author: Jessica Allaire <j_allai@coree.encs.concordia.ca>
Date: Sun Dec 1 15:56:34 2019 -0500
Database mock
commit 13f4deddb8e7622ca661c208b8594cf3bacb8d00
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Sun Dec 1 13:48:22 2019 -0500
Database creation
Added the avatar pictures and SQL commands
commit 31d9d62fec212a575ccc4a4941799099b0f26067 (origin/Developper_Branch)
Merge: 92dd071 bf740c1
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Fri Nov 29 13:21:20 2019 -0500
Merge branch 'Meeting2M' into Developper_Branch
commit bf740c1b0be694e63be10eff681e0f2dc3c78116 (origin/Meeting2M)
Author: Mario Bastiampillai <mariobastiampillai@outlook.com>
Date: Mon Nov 18 03:19:49 2019 -0500
Change of width from pixels to percentage for the submitButton and closeButton, some problems encountered on forms where there's only a closeButton
commit f10754b1f9b3653f19c2d246659a070e3630d3ba
Author: Mario Bastiampillai <mariobastiampillai@outlook.com>
Date: Mon Nov 18 03:06:23 2019 -0500
Link to design.css to for buttons
commit 11232b1decd70ca4a4374b9c9bd57c6d87dd7326 (origin/Popup_Fixes)
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Sun Nov 17 22:33:54 2019 -0500
Finished all the pop ups
commit 058a560d198b23b2a5ba277d21e1f79cb7ab7697
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Sun Nov 17 17:13:46 2019 -0500
For push
commit 2439f92c5eca241012c3c2955ed0fb181d37e0b9
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Wed Nov 13 08:37:56 2019 -0500
Put the JS in a different file, adding the button stuff into design
Also fixed popups for marios stuff
commit 65e7bfa752610024391c5df5ae6609e4e599744c
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Mon Nov 11 15:09:14 2019 -0500
Changed the JS into a different file, modified it to close the popups
commit 0d94498984d6a12d6eeec4989805df53336a0c5e
Author: Mario Bastiampillai <mariobastiampillai@outlook.com>

Date: Mon Nov 11 13:13:06 2019 -0500

    Removal of the green background, borders, etc. Separation of the CSS used to another file.

commit 92dd0718ffd693323e8e784d6ae439374d5ac50b
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Mon Nov 11 12:28:19 2019 -0500

    Fixing merge issues

commit 39f0bffdaf74a2a18fa0e2ff1dabf55e10e3bf4e
Merge: b3d0444 bcfc62e
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Mon Nov 11 11:52:51 2019 -0500

    Merge branch '#20-Event_Manager' into Developper_Branch

commit b3d0444e63cdb55abbe576fbddde27712dc80b0e
Merge: 0b5278d 461d072
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Mon Nov 11 11:50:53 2019 -0500

    Merge branch 'Charles' into Developper_Branch

commit 461d0721d6602640f698931e6c0c06bdce646b3f (origin/m2_user, origin/Charles)
Author: Charles <abouhadiarcharles@gmail.com>
Date: Mon Nov 11 11:48:26 2019 -0500

    by mistake

commit fe1afaf7cef4d186a6f6958ed7c1e0ae2286b1b7
Author: Charles <abouhadiarcharles@gmail.com>
Date: Mon Nov 11 11:39:36 2019 -0500

    fixed image

commit bcfc62ef4db82c10abe9fbcf01862c8d1a1083e5 (origin/#20-Event_Manager)
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Mon Nov 11 11:35:33 2019 -0500

    Added group page

commit 8c86ace83dd011a69ac88d70b68f0cfca32ff1f9
Author: Charles <abouhadiarcharles@gmail.com>
Date: Mon Nov 11 09:54:36 2019 -0500

    Added forms for all 3 user pages

commit 0b5278d710d50ba4e3a19dc0d2de96f5f7da74e9 (origin/Mario)
Author: Mario Bastiampillai <mariobastiampillai@outlook.com>
Date: Mon Nov 11 02:33:17 2019 -0500

    Work on Event page

commit 33fe0a05d5f9acdb1a0b72dd6b8d4e07c347e217
Author: Mario Bastiampillai <mariobastiampillai@outlook.com>
Date: Mon Nov 11 01:50:47 2019 -0500

    Work on the Controller

commit dc5e5774da8ee25be7552b53d05a6603521edb9a
Author: Mario Bastiampillai <mariobastiampillai@outlook.com>
Date: Mon Nov 11 00:42:22 2019 -0500

    Fixing of the positioning and sizing of each form when buttons are clicked

commit 61c9faf68554a633ea446c2a926ccd855baeb490

Author: Mario Bastiampillai <mariobastiampillai@outlook.com>
Date: Sun Nov 10 18:05:06 2019 -0500

Further work on the System Admin page, addition of forms that appear when buttons are clicked on

commit 7b380eb010eb7c34035f1e02702e94e1ff15549f
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Sun Nov 10 16:54:15 2019 -0500

Added event admin homepage and group files

commit aec02b3bc397096e04e608a6c6214dd7228c935f
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Sun Nov 10 16:34:36 2019 -0500

Added Event manager homepage

commit ae3ef3ed27d8507775781f8dc8d42f6b6299cc12
Author: Charles <abouhadiarcharles@gmail.com>
Date: Sun Nov 10 15:56:38 2019 -0500

Added avatars and changed yet again a little bit the layout, and added logout button

commit 3ba391250ed4445c728e4895cb82a04e6bd924ed
Author: Charles <abouhadiarcharles@gmail.com>
Date: Sun Nov 10 01:20:31 2019 -0500

Changed layout of the 3 pages to make it look cleaner, and added some css for the buttons

commit cc523fedb834bea52721f18749bf4c5848734d9a (origin/Developper)
Merge: ae208dc 9eb2d68
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Sat Nov 9 23:29:30 2019 -0500

Merge branch '#9-Header' into Developper

commit 9eb2d68a17a5cadfe01b46060ffc4f4664f2b077 (origin/#9-Header)
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Sat Nov 9 23:28:48 2019 -0500

Added design css

And changed file locations

commit 38709dd1906f0f04fc28d60fb286432d9499b4fd
Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Sat Nov 9 23:28:32 2019 -0500

Added homepage, sign in, sign up, header, footer

commit 85ad1b2b5f9a4367f141e3950e986c07fdfc17c8
Author: Mario Bastiampillai <mariobastiampillai@outlook.com>
Date: Sat Nov 9 20:50:47 2019 -0500

Addition of the Sign Out button on the top right corner, also slight change in color for the buttons

commit 68530db91e743bcf01fa73cd4ebb8bbc98f86bdc
Author: Mario Bastiampillai <mariobastiampillai@outlook.com>
Date: Sat Nov 9 19:53:08 2019 -0500

Work on the System Admin page, including HTML and CSS

commit ae208dc131f3b8170d2260972174cec20ed59642
Merge: ec73b72 4157bba

Author: Jessica Allaire <jessica.allaire.96@hotmail.com>
Date: Sat Nov 9 15:44:33 2019 -0500

    Merge remote-tracking branch 'origin/Charles' into Developper

commit 4157bba703bdeba8364e04dc3d52416882e391c3
Author: Charles <abouhadiarcharles@gmail.com>
Date: Fri Nov 8 18:35:41 2019 -0500

    User home page, settings and profile... just started

commit 9bc57a997ae7714368e7cec7e1edbe579a15b545
Author: Charles <abouhadiarcharles@gmail.com>
Date: Tue Nov 5 15:23:38 2019 -0500

    Added some functions to System Administrator directory

commit 0f908017ec32b28c7821f75a1e46eae0fe084860
Author: Charles <abouhadiarcharles@gmail.com>
Date: Tue Nov 5 14:48:09 2019 -0500

    Front end for Controller, System Administrator, index, Participant ...

commit ccb9bb11cfb96f3c4599f8bf5dc30c7dfc17971d
Author: Charles <abouhadiarcharles@gmail.com>
Date: Mon Oct 21 16:21:48 2019 -0400

    JS

commit 503b35ecf8d3d101bb2c91f7dfb2a53d2408b6e2
Author: Charles <abouhadiarcharles@gmail.com>
Date: Mon Oct 21 06:39:20 2019 -0400

    Website

commit ec73b72a4a88d779f3a20dff1b3e6626cc1b7566 (origin/master, origin/HEAD,
master)
Author: thisischarles <56395727+thisischarles@users.noreply.github.com>
Date: Thu Oct 10 08:14:37 2019 -0400

    Initial commit

(END)

Database

Our database has 29 tables, and this is made to assure that it is in 3NF.

Script to Create the Database and Populate it

Create database

```
CREATE TABLE User (
  UserID int NOT NULL AUTO_INCREMENT PRIMARY KEY,
  LastName varchar(255),
  FirstName varchar(255),
  Password varchar(255),
  Address varchar(255),
  Email varchar(255),
  DOB DATE,
  Username varchar(255),
  Phone bigint,
  ImageID int,
  FOREIGN KEY (ImageID) REFERENCES Images(ImageID),
);
CREATE TABLE Events (
  EventID int NOT NULL AUTO_INCREMENT PRIMARY KEY,
  Duration int(10),
  Name varchar(255),
  Organization varchar(255),
  BandwidthLimit int,
  StorageLimit int,
  CostID int,
  FOREIGN KEY (CostID) REFERENCES Cost(CostID),
  DiscountID int,
  FOREIGN KEY (DiscountID) REFERENCES Discount(DiscountID),
  BandwidthUsed int,
  StorageUsed int,
  AddedBy int,
  FOREIGN KEY (AddedBy) REFERENCES User(UserID),
  Location varchar(255),
  Time Datetime,
  Date Date,
  StatusID int,
  FOREIGN KEY (StatusID) REFERENCES Status(StatusID)
);
CREATE TABLE SystemAdministrator (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID)
);

CREATE TABLE Post (
  PostID int NOT NULL AUTO_INCREMENT PRIMARY KEY,
```

```
  Words varchar(255),
  Author int,
  FOREIGN KEY (Author) REFERENCES User(UserID),
  Duration DATETIME,
  Deleted boolean,
  Media int
);
CREATE TABLE Groupy (
  GroupID int AUTO_INCREMENT PRIMARY KEY,
  Author int,
  FOREIGN KEY (Author) REFERENCES User(UserID),
  Fam boolean NOT NULL,
  Secret boolean NOT NULL,
  Name varchar(255)
);
CREATE TABLE Controller (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID)
  );
CREATE TABLE EventManager (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID)
);
CREATE TABLE EventAdministator (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID)
);
CREATE TABLE EventParticipant (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID)
);
CREATE TABLE GroupParticipant (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
  GroupID int,
  FOREIGN KEY (GroupID) REFERENCES Groupy(GroupID)
);
CREATE TABLE SystemAttributes (
  ChargeRate int,
  ExtraChargeRate int,
```

```
  DefaultBW int,
  DefaultStorage int
 );
CREATE TABLE Status (
  StatusID int AUTO_INCREMENT PRIMARY KEY,
  Status varchar(255)
);
CREATE TABLE Discount (
  DiscountID int AUTO_INCREMENT PRIMARY KEY,
  Discount int
);
CREATE TABLE Cost (
  CostID int AUTO_INCREMENT PRIMARY KEY,
  Cost int
);
CREATE TABLE HasAGroup (
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID),
  GroupID int,
  FOREIGN KEY (GroupID) REFERENCES Groupy(GroupID)
);
CREATE TABLE HasAnEvent (
  GroupID int,
  FOREIGN KEY (GroupID) REFERENCES Groupy(GroupID),
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID)
);
CREATE TABLE Voting (
  VoteID int AUTO_INCREMENT PRIMARY KEY,
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID),
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
  SuggestionID int,
  FOREIGN KEY (SuggestionID) REFERENCES Suggestions(SuggestionID)
);
CREATE TABLE Suggestions (
  SuggestionID int AUTO_INCREMENT PRIMARY KEY,
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID),
  Location varchar(255),
  Date DATE,
  Time DATETIME
);
CREATE TABLE Messages (
  MessageID int AUTO_INCREMENT PRIMARY KEY,
```

```
  Sender int,
  FOREIGN KEY (Sender) REFERENCES User(UserID),
  Message varchar(255),
  New boolean not null,
  MessageStatusID int,
  FOREIGN KEY (MessageStatusID) REFERENCES MessageStatus(MessageStatusID)
);
CREATE TABLE MessageStatus (
  MessageStatusID int not null auto_increment primary key,
  MessageStatus varchar(255)
);
CREATE TABLE Recipient (
MessageID int,
FOREIGN KEY (MessageID) REFERENCES Messages(MessageID),
Recipient int,
FOREIGN KEY (UserID) REFERENCES User(UserID)
);
CREATE TABLE EventTag (
  EventID int,
  FOREIGN KEY (EventID) REFERENCES Events(EventID),
  TagID int,
  FOREIGN KEY (TagID) REFERENCES Tag(TagID)
);
CREATE TABLE UserTag (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
  TagID int,
  FOREIGN KEY (TagID) REFERENCES Tag(TagID)
);
CREATE TABLE GroupTag (
  GroupID int,
  FOREIGN KEY (GroupID) REFERENCES Groupy(GroupID),
  TagID int,
  FOREIGN KEY (TagID) REFERENCES Tag(TagID)
);
CREATE TABLE Tag (
  TagID int AUTO_INCREMENT PRIMARY KEY,
  Tag varchar(255)
);
CREATE TABLE Permissions (
  PermissionID int AUTO_INCREMENT PRIMARY KEY,
  Permission varchar(255)
);
CREATE TABLE HasPermissions (
  UserID int,
  FOREIGN KEY (UserID) REFERENCES User(UserID),
```

```
  PermissionID int,
  FOREIGN KEY (PermissionID) REFERENCES Permissions(PermissionID),
  PostID int,
  FOREIGN KEY (PostID) REFERENCES Post(PostID)
);
CREATE TABLE HasComment (
   PostID int,
   FOREIGN KEY (PostID) REFERENCES Post(PostID),
   CommentID int,
   FOREIGN KEY (CommentID) REFERENCES Post(PostID)
);
CREATE TABLE Images (
   ImageID int AUTO_INCREMENT PRIMARY KEY,
   Image varchar(500)
);
```

Populate database

```
//set user and password credentials for database, and its hostname
$user = 'jrc353_2';
$password = 'gfgaFP';
$database = 'jrc353_2';
//attempt to connect to database
$db = new mysqli("jrc353.encs.concordia.ca", $user, $password, $database) or die("cannot
connect");
//open csv file to populate database
$fin = fopen($_FILES['myFile']['name'],'r') or die('cant open file');
//initialize empty string to then add queries
$theQ = "";
$count = 0; //variable

set_time_limit(9999);

//reading each line of the data file
while (($data=fgetcsv($fin, 5000))!== FALSE) {
   //if the line is the first, skip
   if($count == 0) {
      $count++;
   }
   else {
      $filename = 'Images\Avatar' . $count . ".png";
      $num1 = mt_rand(1, $count); //sender and recipient
      $num2 = mt_rand(1, $count);
      $password = str_replace("\\", "#", crypt($data[8], 'TBD'));
      $phone = (float) str_replace('-', '', $data[7]);
      $theQ = $theQ." INSERT INTO Images (Image) VALUES ('$data[10]');";
      $theQ = $theQ." INSERT INTO User (LastName, FirstName, Password, Address,
Email, DOB, Username, Phone, ImageID) VALUES
('$data[2]','$data[1]','$password','$data[4]','$data[3]','$data[6]', '$data[5]', $phone , $count);";
```

```
        $theQ = $theQ." INSERT INTO Tag (Tag) VALUES ('$data[11]');";
        $theQ = $theQ." INSERT INTO Messages (Sender, Message) VALUES ($num1 ,
'$data[12]');";
        $theQ = $theQ." INSERT INTO Recipient (MessageID, Recipient) VALUES ($count ,
$num2);";
        $count++;
    }
}
//query the whole string of inserts
if($db->multi_query($theQ))
        echo "WORKS";
```

References

www.mockaroo.com
https://www.guru99.com/database-normalization.html
https://www.databasestar.com/database-normalization/