CISC 332:
Course Project Deliverable 3
Final Report

Group 75:
Brandon Tang (10193811)
Tong Chen (10189689)
Will Xie (10189120)

Submitted: April 8 2019

Table of Contents

Assumptions

This list of assumptions made during the development of the application is below. Any additional functionalities that were added by our team will also be listed here.

- The functional requirement “list the schedule…” is asking to list the events available on a particular day, ignoring conflicting times.
    - Individualized schedules for attendees do not exist.
- Level of sponsorship refers to a company’s tier: Platinum, Gold, Silver, Bronze.
- A bed can only fit two people at most.
    - A room with 1 bed has a maximum occupancy of 2 and a room with 2 beds has a maximum occupancy of 4 people.
- Occupancy in room means current occupancy.
    - The maximum occupancy is specified by the previous assumption where 1 bed means a max occupancy of 2, and 2 beds is a max occupancy of 4.
- If new attendee is created and no rooms available, attendee will not be assigned a room and error message will pop up.
    - Rooms will be automatically assigned based on availability. User will not have to select specific room numbers.
    - All rooms in the database will have occupancy of at least 2. Therefore, once rooms are full there will not be any additional rooms added.
- An email attribute will be added to distinguish between attendees. Each attendee will have a unique email address.
- SessionID is associated with a room name, date, and time slot.
- The conference has a limited number of available session rooms and time slots in a given day.
- Event names are unique.
- New sponsor attendees will be added only if company exists in database.
- Attending the conference for free will be implies a rate of $0.
- Job postings are assigned a unique eight-digit Job ID to be distinguishable.
- Job postings are assumed to be only within Canada.
- The maximum pay rate for a job ad will be within the size of what an int type can store.
- Conference sessions are always 2 hours in duration.
- Conference sessions will not be booked if it is in conflict with another session.

ER Diagram:

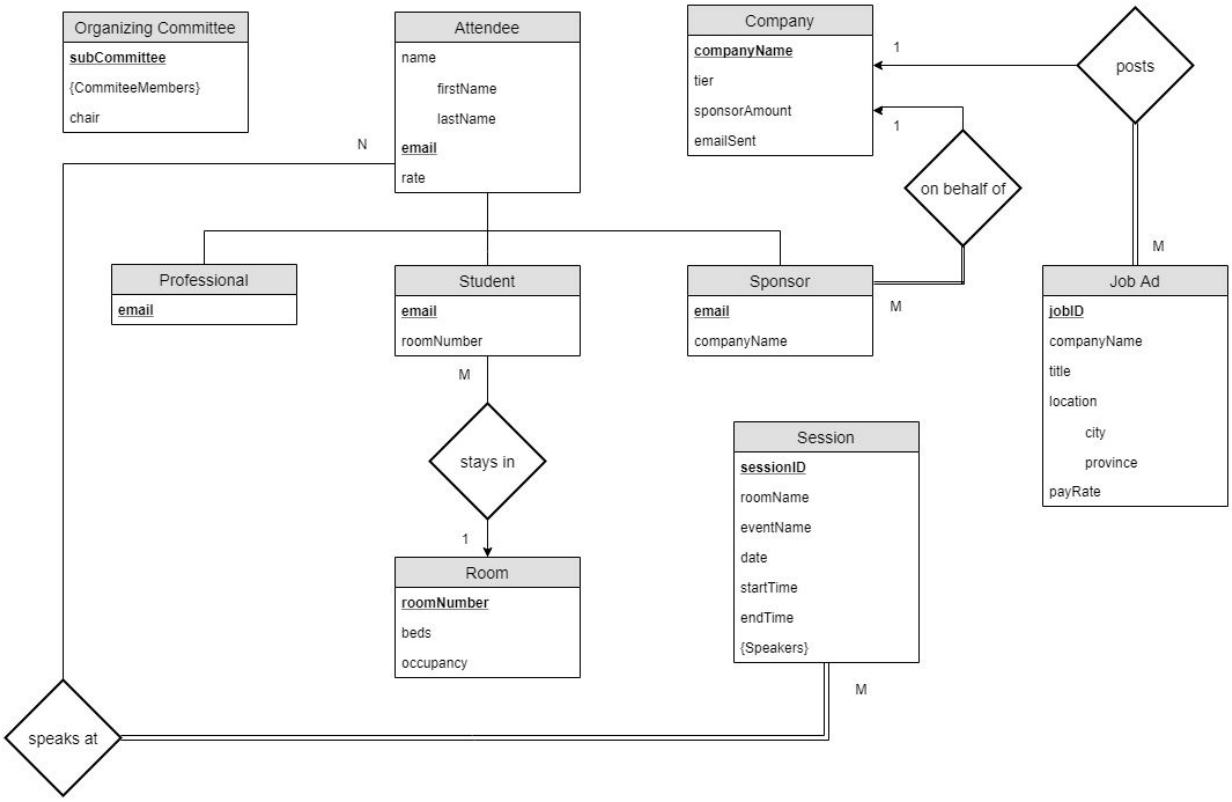


Figure 1. Entity-Relationship (ER) diagram for the database. Rectangular boxes represent tables and diamonds represent relations on tables. Total participation is represented by a double line and all other single lines represent partial participation. Primary keys are bolded and underlined.The characters “1”, “M”, and “N” are used to specify one-to-many and many-to-many relationships.

Discussion

Development Problems and Solutions

We encountered a few problems during the development process however we were able to overcome them without much difficulty.

The first problem encountered was during the requirements analysis phase. The project description had requirements that were ambiguous and lacking detail. This made it difficult for us to determine exactly what was expected in terms of functional and nonfunctional requirements. Since the project allowed us to make assumptions, we were able to resolve the issue surrounding ambiguity by stating assumptions made regarding requirements. The assumptions made were based on previous experiences in software development and looking at the function of the application from a high level. This allowed us to make assumptions about the requirements that would create an application that would function properly, be verseatile and satisfy the needs of our users.

Another problem our team encountered was regarding the creation of our initial entity-relationship (ER) diagram which would lay the foundation for our database. This partially relates to our first problem with requirements analysis because a lot of confusion stemmed from misinterpreting the requirements. However once the requirements were clarified, our team members each had a different view in mind for the ER diagram. The disagreements were regarding the creation of relations, participation of relations, and cardinality of relations. We were able to come to an agreement by laying out all of our options and discussing them thoroughly. This allowed us to analyze the pros and cons of each database design. We acknowledged that each pro had its associated con and decided to take pros that were the most valuable to the database as a whole. Our finalized ER diagram was a collaborative effort where pros from each design were able to be included.

The third problem encountered during the development process was related to the development team. Since our team consisted of three students with varying schedules, it was difficult for us to complete the entire project together. We had to split and delegate tasks to members of the team however this resulted in inconsistencies in work submitted by each member. This was an extensive issue during the development and testing phases due to differences in code. Most of our bugs were a result of differences in code. This ranged from conflicting variable names to functions that did not work as planned. This problem was resolved by setting coding standards to be followed and further breaking down requirements. This allowed us to determine how each requirement would be satisfied in the code, and how each piece would fit together to create the whole application. This resulted in our development and testing phase being much more streamlined as each team member would be able to understand code written by others and combine pieces so that they do not conflict.

Design and Implementation Decisions

The first step to creating our application was deciding on the ER diagram that would lay the foundation for our database. Each team member had their own proposed ER diagram and we considered each of them thoroughly. The finalized ER diagram, shown in Figure 1, takes the pros that we determined to be the most important and beneficial, while minimizing the associated cons. The creation of tables and their associated attributes was fairly consistent for each member of the team. The major decisions made regarding the ER diagram were related to the relations, their cardinality, and participation. The speaks-at relation was created to allow attendees to be assigned a session to speak at, which was one of the requirements. The participation was determined to be partial on the attendee side and total on the session side since not all attendees will be speaking but all sessions will have at least one speaker. The cardinality was determined to be many-to-many.

The on-behalf-of relation allows sponsors, which are a type of attendee, to attend the conference while representing their company. The relation connects sponsors to companies. The participation was determined to be partial on the company side and total on the sponsor side because every sponsor attending must belong to some company but not every company will have sponsors. The cardinality for the on-behalf-of relation is one-to-many.

The stays-in relation allows students to be assigned a room to stay in. The relation connects the student and room tables in the database. The participation of the stays-in relation was determined to be partial on both sides and the cardinality is many-to-one. This is because many students can be assigned to stay in a particular room but the opposite does not hold. Only one room can be assigned to each student based on the requirements.

The posts relation allows companies to post job advertisements at the conference. In the database, this is represented by connecting the company and job ad tables using the posts relation. The participation is partial on the company side and total on the job ad side while the cardinality is one-to-many because each company can post multiple advertisements but a given advertisement cannot belong to multiple companies based on assumptions our team made.

Throughout the development process, user interactions were considered and prioritized. It was important to us that our implementation resulted in the most user friendly application possible. A major design decision, related to user interaction and user experience, was the implementation of the web interface. Our team wanted to improve on the sample code given to us in class. The sample code allowed the user to input values on a web page and submitting the values would navigate to a new web page, which displayed the results. Our implementation allows the user to input and view the results on the same page. This allows the user to make additional queries without having to navigate back to the previous page. This improvement was implemented by creating a template file that contains code for the web page's navigation menu and header. Each file of the web application includes a reference to the template file. This

creates the illusion of a single-page application, with the template file acting as the main container.

Another user interface decision we made was the use drop-down lists where possible. The drop-down lists either had fixed options, in the case of selecting a sponsorship level, or options based upon existing data in the database such as when selecting a particular company to delete. The decision to include drop-down lists provides an element of error proofing as well because the only values displayed that users may select from are valid values. This eliminates the potential for users to select invalid options that would result in an error. Additionally, the project description stated that we could assume user input will be valid however our team decided to implement error checking to ensure our application was be as robust as possible. This was done in PHP using if-statements. The majority of error checking implemented checks for cases where users submit data that has not yet been selected. If an error was encountered, the code in PHP would return an error message to the user informing them of what had gone wrong. The user would be given an opportunity to re-enter their data and try again.

Another major decision made was related to implementing the function to switch a session's day/time and/or location. Our team interpreted the requirement as switching a session with another session. We assumed that there were a limited number of rooms and time slots in a day. This would mean that we could not change the date, time, or location of a single session as it would cause conflicts. In order to change a session’s date, time, or location, our team decided to switch the session of interest with another session. To implement this functionality, we added a sessionID attribute to the Session table. This attribute is meant for backend use only, as system administrators should be the only ones able to switch speaker sessions. By having the sessionID attribute as the primary key and having an assumption that the sessionID is associated with a specific date, location, and time, sessions could easily be switched by swapping the event names and pairing them with the corresponding sessionID.

The last major design decision pertains to the organization of the project files. We followed the software development life cycle during the development of the application and a crucial part of that is maintenance. To ensure our application is maintainable, we decided to combine our HTML and PHP code for each function is a single file. This resulted in fewer files for developers to manage. That is, instead of having an HTML and PHP file for each function, there is exactly half the amount since they are combined. Grouping the code in a single file also helps with traceability and debugging as developers will not need to search between multiple files.

Technologies and Tools Used

- HTML was used to build the framework for our webpage. We used HTML to create headers, paragraphs, and tables to display information to users. Our implementation combined HTML and PHP code to create web pages that could both display and process data.
- CSS was used to format the layout of our web page and improve the presentation of display elements to enhance user experience. We used an external style sheet as this encouraged code re-use and kept format and styling consistent between all of our pages.
- PHP was used to allow HTML code to perform actions upon submission of form data using POST actions. PHP was also used to communicate with MySQL for queries through the use of PDOs.
- MySQL was used to manage our database. Queries were run to create and populate the database. SQL queries could also be tested in MySQL before being implemented in our PHP scripts. The database was also manually tested to verify functionality after implementation, prior to the development of the web application.
- XAMPP was used run our web application and database. XAMPP contains several tools however the specific ones we used were MySQL and Apache. MySQL allowed us to locally host our database and Apache allowed us to locally host a web server so we could properly develop, test, and run our web application.

Potential Improvements

Although our application satisfies the required functionality, it is still very simple in its design and implementation; which leaves a lot of room for improvement. Changes to the overall appearance of our application through adding more CSS elements and including more visuals to fill white space could make our web-page more visually appealing. A re-design of our navigation menu could also provide a more logical order for functionalities, which would enhance the user experience.

From a developer's perspective, our HTML code and PHP code could have be written in different files to separate the presentation from the logic. This would have required further analysis to determine what would be more valuable long term. Increased efficiency of our code could also be made by removing duplicate code and refining already existing code. Encouraging the re-use of code would be a huge improvement and this is something we would definitely do if given more time. Additional detailed documentation could also be added in order to better explain the purpose of certain lines of code. Extra confirmation prompts could be added when data is being deleted or updated to reduce accidental changes to the database. Increased security by requiring a username and password to access the database would also be a significant improvement. This would ensure that only the intended users, which in our case are the organizers, are able to access the database which stores valuable information.

User Guide

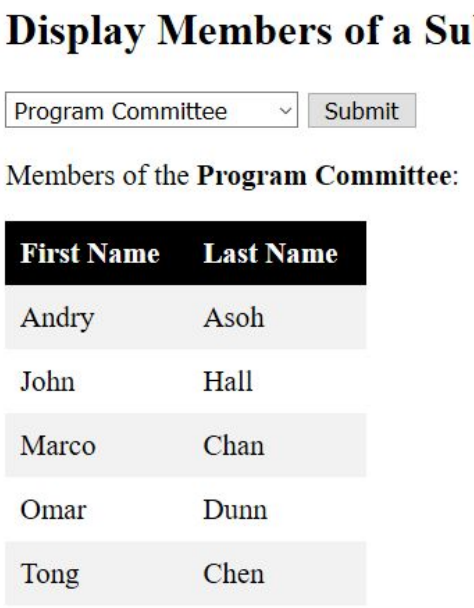
The following is a guide that is meant to instruct you on how to use the database web application. Each function title is bolded. Each function has numbered steps and corresponding instructions that provide details on how to use the function.

A navigation menu is available below to allow you to quickly find your function of choice.

Getting all members of a organizing sub-committee:

1. Using the dropdown menu, select the sub-committee to view and press the submit button.

2. The first and last names of the sub-committee will be displayed.



Getting the total intake from the conference:

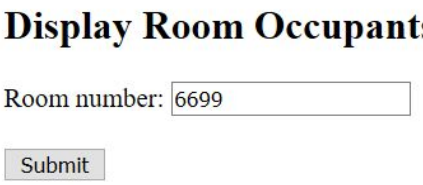
1. Selecting the Total Intake option will automatically display the total intake of the conference, separated by intake from registration fees and sponsorships.

Total Intake by Registration and Sponsorships:

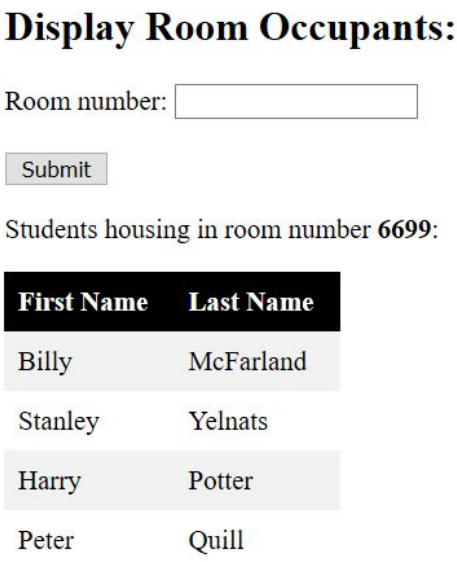
Registration	Sponsorship	Total
1300	34000	35300

Getting the names of students staying in a certain room:

1. Input the selected room number into the text box and press the submit button.



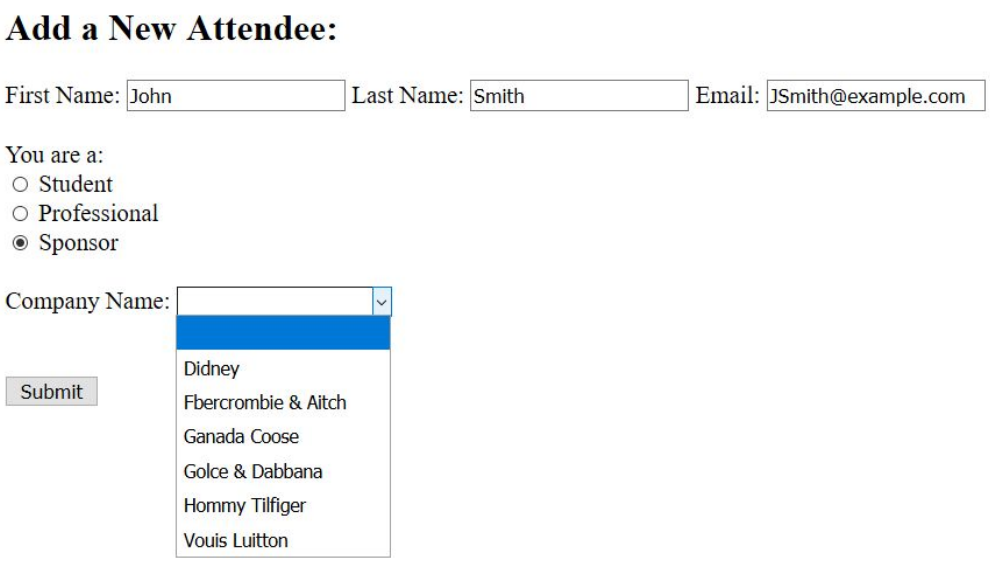
2. The first and last names of the students staying in that room will be displayed.



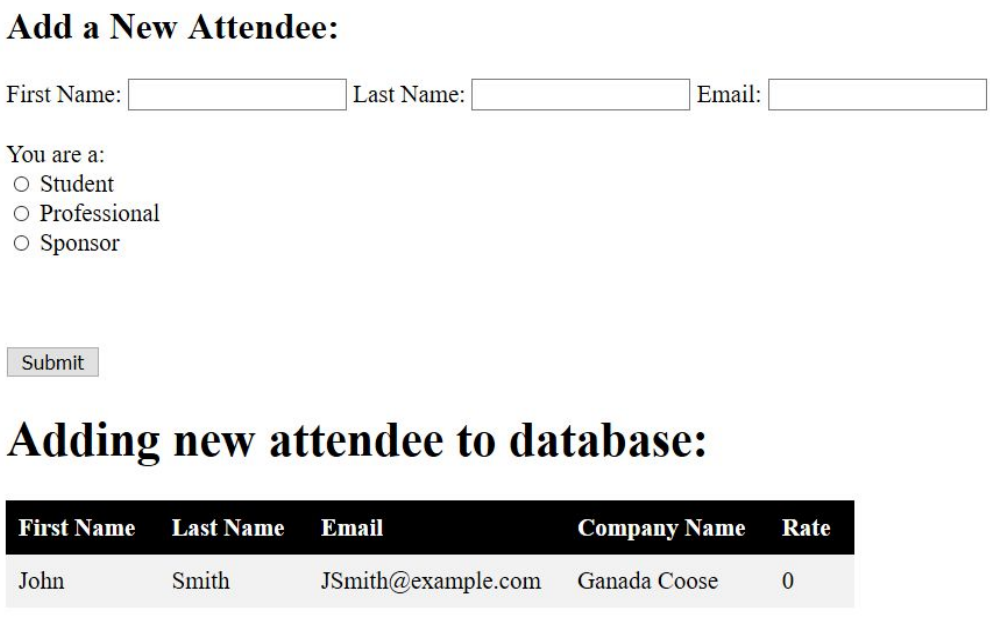
First Name	Last Name
Billy	McFarland
Stanley	Yelnats
Harry	Potter
Peter	Quill

Adding a new attendee to the database:

1. Input the attendee's first name, last name, and email into the text boxes and press the submit button.
   1.1. When adding a sponsor, company name must also be provided.
   1.2. When adding a student, a room will be automatically assigned.



2. The entry of the newly added attendee will be displayed.



Getting the list of conference attendees:

1. Selecting the Conference Attendees option will automatically display all the attendees, seperated into lists of students, professionals, and sponsors.

Display all Attendees by Groups:

List of **Student** attendees:

First Name	Last Name	Email
Billy	McFarland	bestfestival@fyre.com
Jake	Peralta	brooklyn99@nbc.com
Stanley	Yelnats	holes@realmail.com
Harry	Potter	ilive@realmail.com
Joseph	Park	j.park@realmail.com
Simba	Lion	ripdad@realmail.com
Simon	Zheng	s.zheng@realmail.com
Peter	Quill	shame@avengers.com
Peter	Parker	spidurman@avengers.com
Kanye	West	ye@realmail.com

List of **Professional** attendees:

First Name	Last Name	Email
Twentyone	Savage	deported@ice.com
Kyle	Lee	k.lee@realmail.com
Tony	Stark	stark@avengers.com
Spongebob	Squarepants	sweetvictory@superbowl.us
Ash	Ketchum	verybest@everwas.com
Yugi	Mutou	yugi@oh.com

List of **Sponsor** attendees:

First Name	Last Name	Email
Ronald	McDonald	bigmac@mcdonalds.com
Kanye	West	imaletyoufinish@taylor.com
Kyle	Lee	k.lee@realmail.com
Tony	Stark	stark@avengers.com
Bill	Nye	thescience@guy.com
Yugi	Mutou	yugi@oh.com

Getting the list of all sponsoring companies:

1. Selecting the Sponsor Companies option will automatically display all the sponsoring companies and their sponsorship tier.

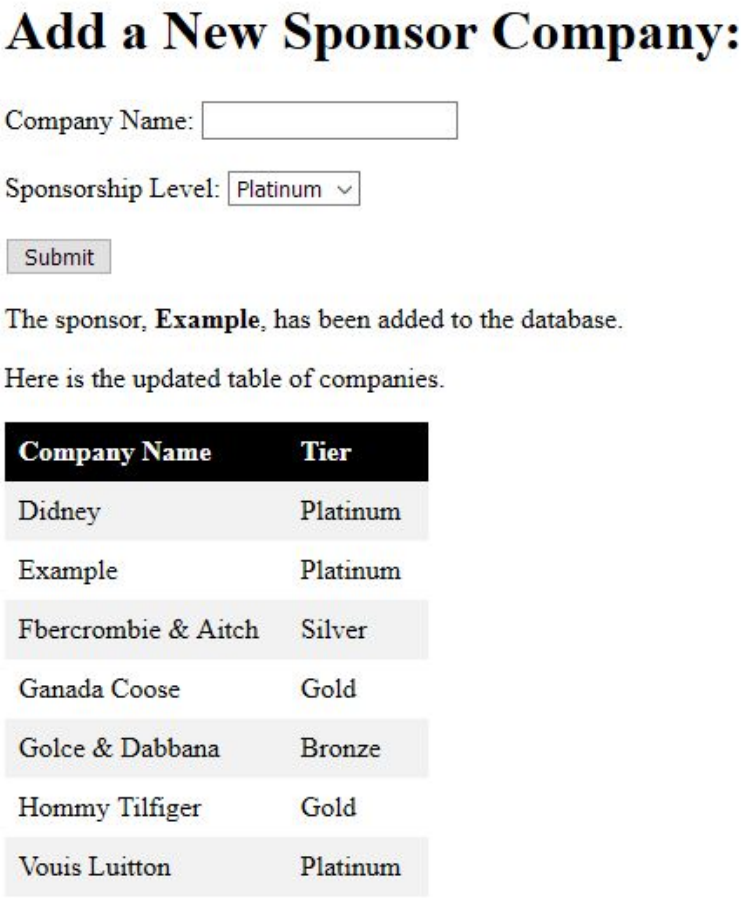
Sponsor Companies and Sponsorship Levels:

Company Name	Tier
Didney	Platinum
Fbercrombie & Aitch	Silver
Ganada Coose	Gold
Golce & Dabbana	Bronze
Hommy Tilfiger	Gold
Vouis Luitton	Platinum

Adding a new sponsoring company to the database:

1. Input the company name into the text box and select the sponsorship level from the dropdown menu and press the submit button.

2. The updated table of sponsoring companies will be displayed.

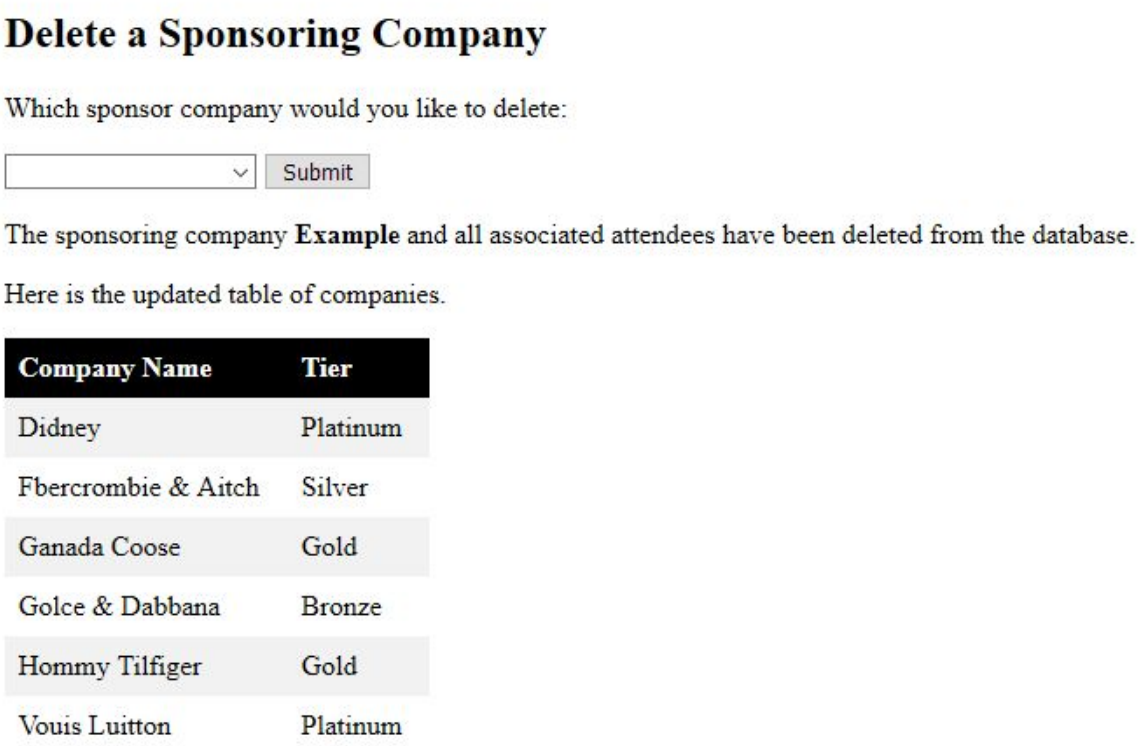


Company Name	Tier
Didney	Platinum
Example	Platinum
Fbercrombie & Aitch	Silver
Ganada Coose	Gold
Golce & Dabbana	Bronze
Hommy Tilfiger	Gold
Vouis Luitton	Platinum

Deleting a sponsoring company and its associated attendees:

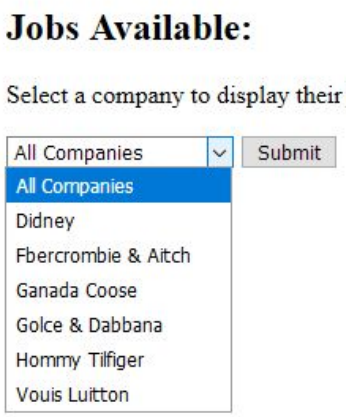
1. Select the company to be deleted from the dropdown menu and press the submit button.

2. The updated table of sponsoring companies will be displayed.



Getting job postings from companies:

1. Select the company to view job postings for from the dropdown menu, or select All Companies to view all job postings and press the submit button.

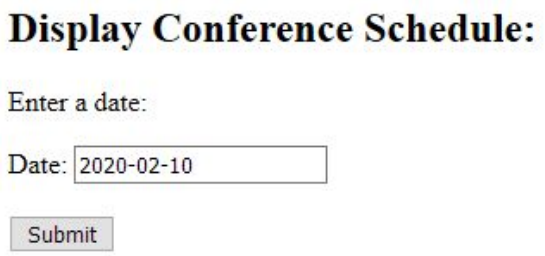


2. A list of job postings with associated information will be displayed.

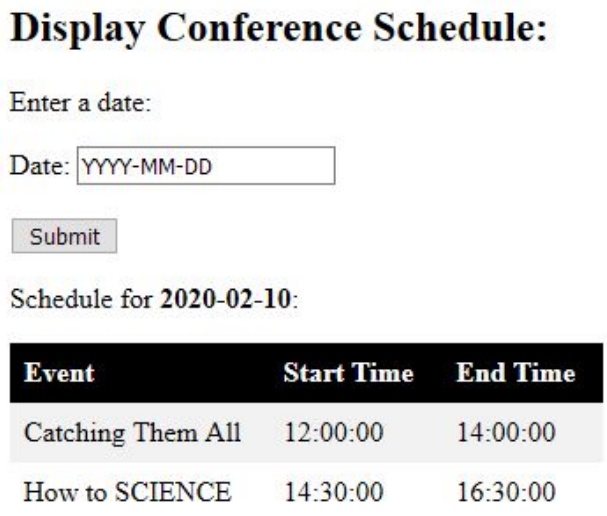
Company Name	Job Title	Job ID	Province	City	Payrate
Golce & Dabbana	Public Relations Consultant	12634967	Ontario	Kingston	12000
Fbercrombie & Aitch	CEO	42843653	British Columbia	Vancouver	190000
Hommy Tilfiger	Store Manager	69291783	Manitoba	Winnipeg	24000
Ganada Coose	Sales Representative	73957103	Alberta	Calgary	87000
Vouis Luitton	Cash Counter	73982094	Quebec	Montreal	207000
Didney	Marketing Associate	93656103	Ontario	Toronto	34000

Getting the conference schedule for a particular day:

1. Input the date to get schedule for into the text box in the format YYYY-MM-DD and press the submit button.

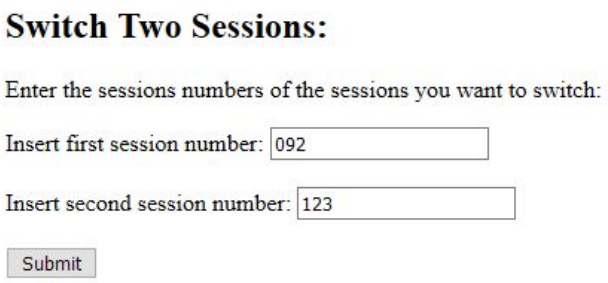


2. All events for that day will be displayed, along with start and end times.

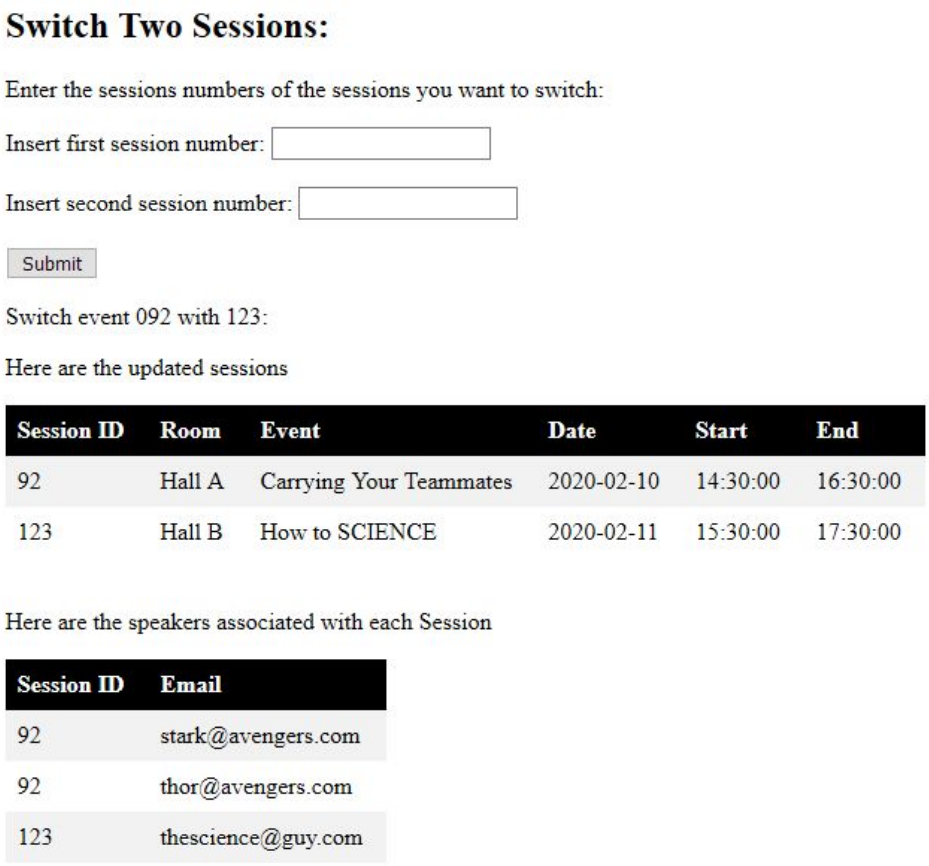


Swapping session time and location:

1. Input the session numbers of the two sessions to be swapped and press the submit button.



2. Updated lists of the session times and locations along with the associated speakers will be displayed.



PHP File (for the Add Sponsors function):

```
<?php
  include("template.php");
?>

<div class="contentContainer">
      <h1> Add a New Sponsor Company: <br> </h1>

      <form method="post">
         Company Name:
         <input type="text" name="companyName"><br><br>

         Sponsorship Level:
         <select name = "tier">
            <option value="Platinum">Platinum</option>
                  <option value="Gold">Gold</option>
                  <option value="Silver">Silver</option>
                  <option value="Bronze">Bronze</option>
         </select> <br><br>
         <input type="submit" name="submit" value="Submit"/>
      </form>

      <?php
      if ($_SERVER['REQUEST_METHOD'] == "POST") {
         $company = $_POST['companyName'];  // Storing Selected Value In Variable
         $tier = $_POST['tier'];  // Storing Selected Value In Variable
         // $amount = "";
         // $numEmails = "";

         switch ($tier) {
            case "Platinum":
               $amount = '10000';
               $numEmails = '5';
               break;

            case "Gold":
               $amount = '5000';
               $numEmails = '4';
               break;

            case "Silver":
               $amount = '3000';
               $numEmails = '3';
               break;

            case "Bronze":
```

```
            $amount = '1000';
            $numEmails = '0';
            break;
        }

        #connect to the database
        $pdo = new PDO('mysql:host=localhost;dbname=conferencedb', "root", "");

        # create and execute query
        $sql = "insert ignore into Company values (?, ?, ?, ?)";
        $stmt = $pdo->prepare($sql);
        $stmt->execute(array($company, $tier, $amount, $numEmails));

        # error-checking
        if ($stmt->rowCount() == 0) {
            echo "<p>The sponsor you are trying to add already exists in the database.</p>";
        } else {
            echo "<p>The sponsor, <strong>$company</strong>, has been added to the
database.</p>";

            # create and execute query
            $sql2 = "select companyName, tier from company";
            $stmt = $pdo->prepare($sql2);
            $stmt->execute();

            # display results in table
            echo "<p>Here is the updated table of companies.</p>";
            echo "<table><tr><th>Company Name</th><th>Tier</th></tr>";

            while ($row = $stmt->fetch()) {
                echo "<tr><td>".$row["companyName"]."</td><td>".$row["tier"]."</td></tr>";
            }
        }
    }

    ?>
</div>
```

Script to Build and Populate Database:

```
create table OrganizingCommittee
        (subCommittee                   varchar(30) not null,
         chair                          varchar(40) not null,
         primary key (subCommittee)
         );

create table CommitteeMembers
        (subCommittee                   varchar(30) not null,
         firstName                      varchar(20) not null,
         lastName                       varchar(20) not null,
         email                  varchar(40) not null,
         primary key (subCommittee, email),
         foreign key (subCommittee) references OrganizingCommittee (subCommittee)
           on delete cascade
         );

create table Attendee
        (firstName              varchar(20) not null,
         lastName               varchar(20) not null,
         email              varchar(40) not null,
         rate          enum('0', '50', '100'),
         primary key (email)
         );

create table Professional
        (email      varchar(40) not null,
         primary key (email),
         foreign key (email) references Attendee (email)
       on delete cascade
    );

create table Room
        (roomNumber             int not null,
         beds                   enum('1', '2') not null,
         occupancy              enum('2', '3', '4') not null,
         primary key (roomNumber)
         );

create table Student
        (email              varchar(40) not null,
```

```
        roomNumber     int,
        primary key (email),
        foreign key (roomNumber) references Room (roomNumber)
      on delete set null,
        foreign key (email) references Attendee (email)
      on delete cascade
    );

create table Company
        (companyName       varchar(40) not null,
        tier                    enum('Platinum', 'Gold', 'Silver', 'Bronze') not null,
        sponsorAmount     enum('10000', '5000', '3000', '1000') not null,
    emailSent        int,
    primary key (companyName)
        );

create table Sponsor
        (email                varchar(40) not null,
        companyName          varchar(30) not null,
        primary key (email),
        foreign key (companyName) references Company (companyName)
      on delete cascade,
        foreign key (email) references Attendee (email)
      on delete cascade
        );

create table Session
        (sessionID      int not null,
    roomName            varchar(30) not null,
        eventName               varchar(30),
        date                    date not null,
        startTime               time not null,
        endTime                 time,
        primary key (sessionID)
        );

create table Speakers
        (sessionID      int not null,
        email                   varchar(40) not null,
        primary key (sessionID, email),
        foreign key (email) references Attendee (email)
                on delete cascade,
        foreign key (sessionID) references Session (sessionID)      on delete cascade
```

```
        );

create table JobAd
        (jobID                  char(8) not null,
         companyName            varchar(40) not null,
         title          varchar(30) not null,
         city                   varchar(20) not null,
    province            varchar(20) not null,
    payRate             int,
    primary key (jobID),
    foreign key (companyName) references Company (companyName)
      on delete cascade
    );

insert into OrganizingCommittee values ('Program Committee', 'Brandon Tang');
insert into OrganizingCommittee values ('Registration Committee', 'Tong Chen');
insert into OrganizingCommittee values ('Sponsorship Committee', 'Will Xie');

insert into CommitteeMembers values ('Program Committee', 'Tong', 'Chen',
'tongtongtong@hotmail.com');
insert into CommitteeMembers values ('Program Committee', 'Andry', 'Asoh',
'andryasoh@hotmail.com');
insert into CommitteeMembers values ('Program Committee', 'Marco', 'Chan',
'lilmarco@gmail.com');
insert into CommitteeMembers values ('Program Committee', 'Omar', 'Dunn',
'od@realmail.com');
insert into CommitteeMembers values ('Program Committee', 'John', 'Hall',
'j.hall@realmail.com');
insert into CommitteeMembers values ('Registration Committee', 'Brandon', 'Tang',
'btangizzle@yahoo.com');
insert into CommitteeMembers values ('Registration Committee', 'Gary', 'Liang',
'garyhdnwliang@live.com');
insert into CommitteeMembers values ('Registration Committee', 'Junlong', 'Yang',
'jundong@hotmail.com');
insert into CommitteeMembers values ('Registration Committee', 'Eugene', 'Lloyd',
'e.lloyd@realmail.com');
insert into CommitteeMembers values ('Registration Committee', 'Leon', 'Lawrence',
'l.law@realmail.com');
insert into CommitteeMembers values ('Sponsorship Committee', 'Will', 'Xie',
'wetwillie@yahoo.com');
insert into CommitteeMembers values ('Sponsorship Committee', 'Bill', 'Xie',
'billygoat@yahoo.com');
```

```
insert into CommitteeMembers values ('Sponsorship Committee', 'Will', 'Smith',
'AhhhhThatsHot@youtube.com');
insert into CommitteeMembers values ('Sponsorship Committee', 'Chi', 'Gieng',
'chizee.chi@realmail.com');
insert into CommitteeMembers values ('Sponsorship Committee', 'Eddie', 'Pham',
'fam@realmail.com');

insert into Attendee values ('Spongebob', 'Squarepants', 'sweetvictory@superbowl.us', '100');
insert into Attendee values ('Twentyone', 'Savage', 'deported@ice.com', '100');
insert into Attendee values ('Ash', 'Ketchum', 'verybest@everwas.com', '100');
insert into Attendee values ('Bill', 'Nye', 'thescience@guy.com', '0');
insert into Attendee values ('Ronald', 'McDonald', 'bigmac@mcdonalds.com', '0');
insert into Attendee values ('Kanye', 'West', 'imaletyoufinish@taylor.com', '0');
insert into Attendee values ('Jake', 'Peralta', 'brooklyn99@nbc.com', '50');
insert into Attendee values ('Peter', 'Parker', 'spidurman@avengers.com', '50');
insert into Attendee values ('Billy', 'McFarland', 'bestfestival@fyre.com', '50');
insert into Attendee values ('Peter', 'Quill', 'shame@avengers.com', '50');
insert into Attendee values ('Thor', 'Odinson', 'thor@avengers.com', '100');
insert into Attendee values ('Tony', 'Stark', 'stark@avengers.com', '0');
insert into Attendee values ('Yugi', 'Mutou', 'yugi@oh.com', '0');
insert into Attendee values ('Jill', 'Richards', 'j.rich@realmail.com', '100');
insert into Attendee values ('Homer', 'Wang', 'h.wang@realmail.com', '100');
insert into Attendee values ('Albert', 'Rivera', 'a.river@realmail.com', '100');
insert into Attendee values ('Darren', 'Wolfe', 'd.wolfe@realmail.com', '100');
insert into Attendee values ('Simon', 'Zheng', 's.zheng@realmail.com', '50');
insert into Attendee values ('Joseph', 'Park', 'j.park@realmail.com', '50');
insert into Attendee values ('Kyle', 'Lee', 'k.lee@realmail.com', '0');
insert into Attendee values ('Simba', 'Lion', 'ripdad@realmail.com', '50');
insert into Attendee values ('Stanley', 'Yelnats', 'holes@realmail.com', '50');
insert into Attendee values ('Kanye', 'West', 'ye@realmail.com', '50');
insert into Attendee values ('Harry', 'Potter', 'ilive@realmail.com', '50');
insert into Attendee values ('Muston', 'Aatthews', 'am@leafs.ca', '50');
insert into Attendee values ('Mitch', 'Marner', 'mm@leafs.ca', '50');
insert into Attendee values ('Bom', 'Trady', 'deflate@gate.com', '50');
insert into Attendee values ('JeBron', 'Lames', 'goodbye@cavs.com', '50');
insert into Attendee values ('Tustin', 'Jrudeau', 'pm@canada.ca', '50');
insert into Attendee values ('Tonald', 'Drump', 'cantstump@thetrump.com', '50');

insert into Professional values ('sweetvictory@superbowl.us');
insert into Professional values ('deported@ice.com');
insert into Professional values ('verybest@everwas.com');
insert into Professional values ('stark@avengers.com');
insert into Professional values ('yugi@oh.com');
```

```
insert into Professional values ('k.lee@realmail.com');

insert into Room values (1738, '1', '2');
insert into Room values (6699, '2', '4');
insert into Room values (7341, '2', '3');
insert into Room values (1111, '2', '2');
insert into Room values (2222, '2', '2');
insert into Room values (3333, '2', '2');

insert into Student values ('brooklyn99@nbc.com', 1738);
insert into Student values ('spidurman@avengers.com', null);
insert into Student values ('bestfestival@fyre.com', 6699);
insert into Student values ('shame@avengers.com', 6699);
insert into Student values ('s.zheng@realmail.com', 7341);
insert into Student values ('j.park@realmail.com', 7341);
insert into Student values ('ripdad@realmail.com', 1738);
insert into Student values ('holes@realmail.com', '6699');
insert into Student values ('ye@realmail.com', '7341');
insert into Student values ('ilive@realmail.com', '6699');
insert into Student values ('am@leafs.ca', '1111');
insert into Student values ('cantstump@thetrump.com', '1111');
insert into Student values ('mm@leafs.ca', '2222');
insert into Student values ('goodbye@cavs.com', '2222');
insert into Student values ('pm@canada.ca', '3333');
insert into Student values ('deflate@gate.com', '3333');

insert into Company values ('Ganada Coose', 'Gold', '5000', '4');
insert into Company values ('Didney', 'Platinum', '10000', '5');
insert into Company values ('Golce & Dabbana', 'Bronze', '1000', '0');
insert into Company values ('Vouis Luitton', 'Platinum', '10000', '5');
insert into Company values ('Hommy Tilfiger', 'Gold', '5000', '4');
insert into Company values ('Fbercrombie & Aitch', 'Silver', '3000', '3');

insert into Sponsor values ('thescience@guy.com', 'Didney');
insert into Sponsor values ('imaletyoufinish@taylor.com', 'Ganada Coose');
insert into Sponsor values ('bigmac@mcdonalds.com', 'Golce & Dabbana');
insert into Sponsor values ('stark@avengers.com', 'Vouis Luitton');
insert into Sponsor values ('k.lee@realmail.com', 'Hommy Tilfiger');
insert into Sponsor values ('yugi@oh.com', 'Fbercrombie & Aitch');

insert into Session values (647, 'Hall A', 'Catching Them All', '2020-02-10', '12:00:00',
'14:00:00');
insert into Session values (092, 'Hall A', 'How to SCIENCE', '2020-02-10', '14:30:00', '16:30:00');
```

```
insert into Session values (637, 'Hall B', 'I Messed Up', '2020-02-11', '13:00:00', '15:00:00');
insert into Session values (123, 'Hall B', 'Carrying Your Teammates', '2020-02-11', '15:30:00',
'17:30:00');
insert into Session values (192, 'Hall C', 'Heart of the Cards', '2020-02-12', '12:00:00',
'14:00:00');
insert into Session values (917, 'Hall C', 'Safety in NYC', '2020-02-12', '14:30:00', '16:30:00');

insert into Speakers values (647, 'verybest@everwas.com');
insert into Speakers values (092, 'thescience@guy.com');
insert into Speakers values (637, 'shame@avengers.com');
insert into Speakers values (123, 'thor@avengers.com');
insert into Speakers values (123, 'stark@avengers.com');
insert into Speakers values (192, 'yugi@oh.com');
insert into Speakers values (917, 'brooklyn99@nbc.com');
insert into Speakers values (917, 'spidurman@avengers.com');

insert into JobAd values (73957103, 'Ganada Coose', 'Sales Representative', 'Calgary', 'Alberta',
'87000');
insert into JobAd values (93656103, 'Didney', 'Marketing Associate', 'Toronto', 'Ontario',
'34000');
insert into JobAd values (12634967, 'Golce & Dabbana', 'Public Relations Consultant',
'Kingston', 'Ontario', '12000');
insert into JobAd values (73982094, 'Vouis Luitton', 'Cash Counter', 'Montreal', 'Quebec',
'207000');
insert into JobAd values (69291783, 'Hommy Tilfiger', 'Store Manager', 'Winnipeg', 'Manitoba',
'24000');
insert into JobAd values (42843653, 'Fbercrombie & Aitch', 'CEO', 'Vancouver', 'British
Columbia', '190000');
```