PROJECT 1: BUILD A WEB APPLICATION USING PHP

You have been asked to create a web application for an event management system. In order to use the application, the users need to login. There are 3 types of users depending on the permissions: attendee (view permission), event manager (partial control) and admin (full control). Please find other requirements as listed below.

Requirements

- Use the provided SQL script to create the tables on Solace. There is also an EER diagram of the tables for you to review.
- You can add tables if you want (e.g. *permission*), but you cannot modify or delete the tables created by the script.
- You will have to populate the tables with a minimum of:
  - 2 venues, 2 events, 1 session for each event,
  - 2 attendees for each event and the 1 session in each event, and at least 1 attendee of each type (admin, event manager - needs to be assigned to an event, and attendee – needs to be assigned to an event and session
  - passwords need to be hashed using sha256.
- The application will consist of at least 2 pages:
  - **Events** (listing of events with session schedule and venue location; it needs to be visible if the user is registered for an event, and that user should be capable of managing her/his registration),
  - **Admin** (add users, venues, events and sessions).
- Users need to be logged in to use the application (this can be done as a separate page or other method). If the user isn't logged in, you need to require them to login. Also, provide a logout option.
- You will use sessions to control access to the pages based on the roles:

  ATTENDEE ROLE
  - Events Page:
    - View all events.
  - Registrations Page:
    - Select, add, update, delete and view their registrations.

  EVENT MANAGER ROLE
  - Attendee role functionality plus what follows
  - Admin Page:
    - Add/View/Edit/Delete their own events, sessions in their own events, attendees in their own events/schedules

  ADMIN ROLE (need to have one master admin that can't be deleted or edited)
  - All functionality of all other roles, plus what follows
  - Admin Page:
    - Add/Edit/Delete/View any user, venues, events, sessions, attendees

- **Good programming principles and practices applied**
  - CSS should be defined in external style sheets.
  - The site should pass HTML5 validation.
  - All pages share common visual and navigation elements. You may use any front-end technology you wish. The common elements for each page (navigation, etc.) will NOT use input/required statements to include those elements. You need to achieve this via templates or functions.
  - Your code should be structured to be re-usable and easily maintained. This means extensive use of functions and classes.
  - All input must be validated and sanitized on the server-side.
  - All DB queries must be parametrized queries using prepared statements.

GUI Requirements (Visualizing the Data)

Regarding the GUI, the GUI should not be a copy-paste of the DB tables. If you provide the same tables, and in the same format as found in the Database itself, then, there is no point in having the web application. The web application should ease the process of editing the tables found in the database. Otherwise, it would be easier to turn to the MySQL Workbench and edit the DB tables there.

For example, let's consider you are an admin and would like to edit the Events data. As an admin, you do not want to see and edit data that you do not understand as in here (dealing with event's and venue' ids):

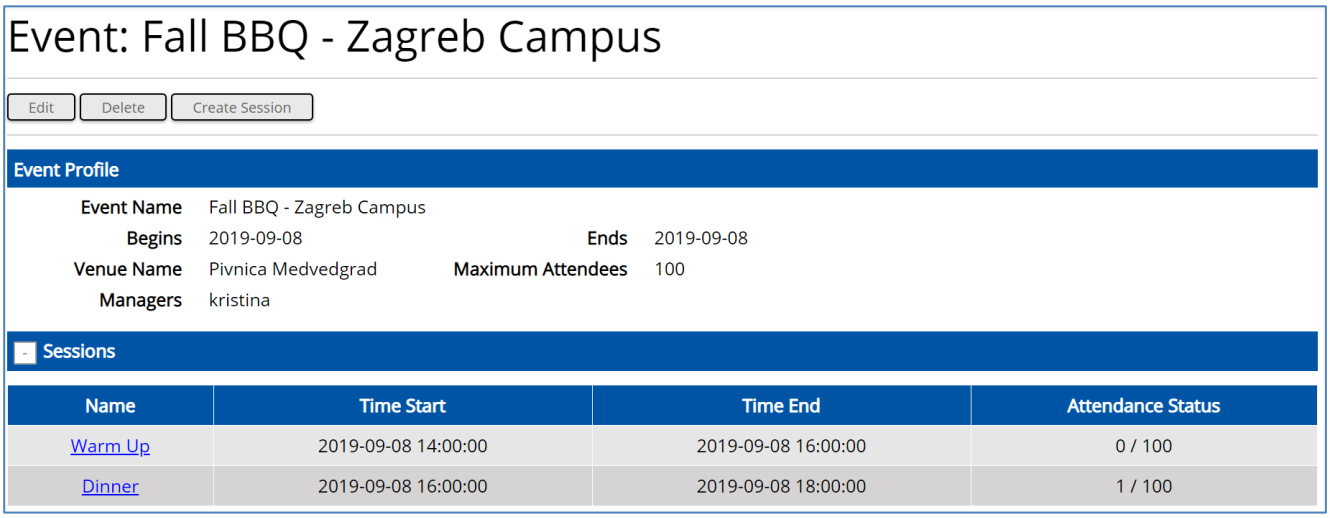
	idevent	name	datestart	dateend	numberallowed	venue
Edit Copy Delete	10	Billy Joel Concert	2019-03-13 00:00:00	2019-03-15 00:00:00	13000	6
Edit Copy Delete	9	Islanders Game	2019-03-06 00:00:00	2019-03-06 00:00:00	1000	2
Edit Copy Delete	11	Fall BBQ - Zagreb Campus	2019-09-08 00:00:00	2019-09-08 00:00:00	100	7

This table is a copy of the DB table. You, as an admin, would like to see a table like this:

Name	Venue	Dates	Sessions
Islanders Game	Nassau Coliseum	03/06/2019 through 03/06/2019	• 03/06/2019 18:00 - '6:00 Game' (3/750)
Billy Joel Concert	Madison Square Garden	03/13/2019 through 03/15/2019	• 03/13/2019 21:00 - '9:00 Showtime' (3/13000)
Fall BBQ - Zagreb Campus	Pivnica Medvedgrad	09/08/2019 through 09/08/2019	• 09/08/2019 14:00 - 'Warm Up' (0/100) • 09/08/2019 16:00 - 'Dinner' (1/100)

where useful data is presented, including sessions as well. This kind of table is possible to create only if you collect data from multiple DB tables, and this combined data is what you need to provide and present to the end users (admin, managers, users).

Once more, if you as an admin want to edit an event, you should be able to update (in one web page) everything that is event related (event and the event's sessions) as in:



FYI: A session is a part of an event. For example, this **Fall BBQ** event has 2 sessions: **Warm Up** & **Dinner**. As a regular user, you may register to the warm up sessions only, and skip the dinner.

Deliverables

- Export your DB tables when you have completed the application to the project's **assets** folder.
- Create a **passwords.txt** file with all usernames and passwords you use and save it to the project's **assets** folder.
- Zip up your entire project (file structure intact) in a folder with your last name and upload it to the dedicated MyCourses Assignments folder.
- Include a link to your application on Solace with all usernames and passwords in the comments section when you upload your project to MyCourses Assignment folder.

- **DUE DATE: Beginning of W06**

Grading

Meeting the above requirements will result in a 'B'. See the rubric for options required to get an 'A'.

	Possible Points	Points Achieved
Proper Functionality	**40**	
Must be logged in and uses sessions to manage	**10**	
Admin role functionality	**15**	
Manager role functionality	**10**	
Attendee role functionality	**5**	
Coding Practices	**50**	
Passes HTML5 validation, CSS in external files	**5**	
Common elements are not included/required but are created using functions and/or templates	**5**	
Functions and classes are used for re-usability and maintainability	**10**	
All input validated	**10**	
All input sanitized	**10**	
All DB queries must be parametrized queries using prepared statements	**5**	
Passwords are hashed using sha256	**5**	
SUBTOTAL	**90**	
Grade A Points (You may earn more than 10 pts)	**25**	
Database functionality abstracted into separate class	**5**	
Database functions use PDO	**5**	
Database functions use PDO with Object Mapping	**5**	
Registration of users	**5**	
Grade A Points Earned:		
Grade Earned (maximum: 100 points)	**100**	