BIRZEIT UNIVERSITY

Computer Science Department

Web Application and Technologies (COMP334)
2nd Semester 2021/2022

Final Project **Due Date: 14-06-2022**

In this final project, you will be converting your last assignment from static pages to dynamic pages using PHP and MariaDB or MySQL. You are forbidden to use any code builder or any external library in this project. You should do the entire project by writing HTML, CSS, PHP and SQL files using a plain text editor like notepad++ or sublime...

General Requirements:

- Move all of your files in the hosting server to a new folder named “backup”.
- You must upload your folders and files, including the SQL and insert statements, to your web hosting server inside the the root `web/` folder, so that the first page that will appear when opening your website will be the login page.
- Also you must compress all of your files, including the SQL and insert statements, to a zipped archive file named `Project_{STUDENT_ID}.zip` and submit it to Ritaj by replaying to the message before the due date.
- If any part of your code or pages are found somewhere else on the Web, or copied from anywhere, or you used any existing library, you'll get a **ZERO** mark.
- Any extra code that do something extra other than what is required in the project or if you used anything that we did not study in this course until now will make you **loose your marks**, so write **only** what is required in this assignment and using the techniques you **only** learned in this course until now.
- We will not accept any submission sent as a memo, and will not accept any submission later than the due date.

Discussion instructions

1. You must book a time slot for your discussion and commit to it. Time slots will be announced later.
2. The Project discussion is considered as an **official exam**, if you miss it you will get an **ZERO** mark.
3. You will have 5 minutes to demonstrate you work on the server. You need to provide a demo of your site all the functionality as specified in the task description.
4. Then we will have 10 minutes for discussion, we will ask you questions all about the course, even if you did not implement in your project. For example: in CSS we might ask you about what are the inherited proprieties, what is the difference between cascade and inheritance.
5. The code discussion will also be depending on the code you have submitted, so you need to know and explain every single word you write in your files.

Introduction:

In this project you will be doing major work on the "**Students Training**" website, and you will be using your code from last assignment as your starting code for this assignment.

First Steps:

In this assignment, we will start by doing the following steps:

1. Creating a Database with the following tables details, including all the references and constraints needed (the SQL should be saved included with the project files):
   - 1.1. **user**: id, username (unique), password, display name, last hit, user type (company or student)
   - 1.2. **city**: id, name, country
   - 1.3. **student**: id, name, city id, email (unique), tel (unique), university, major, projects, interests, photo path, user id
   - 1.4. **company**: id, name, city id, email (unique), tel (unique), positions count, positions details, logo path, user id
   - 1.5. **students_applications**: id, student id, company id, apply date, requested by user id, application status (sent (default), approved, rejected). (unique student id and company id)
2. Add an initial data for the tables as the following (the SQL should be saved included with the project files):
   - 2.1. 1 student user with 1 student record.
   - 2.2. 1 student user without a student record.
   - 2.3. 1 company user with 1 company record.
   - 2.4. 1 company user without a company record.
   - 2.5. Add 5 Palestinians cities to the city table.
3. The user password should be stored in the database using **SHA1**(password) database function, i.e. password should not be stored in plain text and should not be readable in the user table.

Assignment Requirements:

General requirements:

1. Use **only external** CSS files, **NO** embedded or inline CSS are allowed.
2. All CSS files should be valid CSS without any error or warning, you can check this on: https://jigsaw.w3.org/css-validator/
3. All pages should produce valid HTML without any error or warning, you can check this on: https://validator.w3.org/

4. You must put your connection details in a separate file called “parts/_db.php”, it should declare the required variables used to create a connection, and create a PDO object.
5. You should always use **ONLY** prepared statements with named binding parameters for any database SQL.
6. The students photos and the companies logos should be stored in separate 2 folders.

User login:

1. The first page named “index.php” should implement the login for the user, based on the users details stored in “user” table.
2. For logged in users: the login link in the <header> should be changed to “Welcome [display name]” and a logout link next to it.
3. The logout link should go to “logout.php” and it should end the current user session.
4. Each time the user open any page in the website, the “last hit” column of the user table should be updated.
5. The user session should store the user id, display name, and user type, and the user company id or student id if he has one, **or** when he add one.
6. If the user is logged in, he can not see the login form again, instead, his “index.php” page should display information as requested in the last point bellow.

Student pages:

1. The “students.php” page:
   - 1.1. Should list all of the students from the database.
   - 1.2. Should be searchable by any part of the student major (like) or by a specific city, or by both of them.
   - 1.3. The student name should link to the student details page of the clicked student link.
2. The “student.php” page:
   - 2.1. If the current user is the same student in the details page (the user id of the student), he can edit his student details by clicking “Edit” link, otherwise no edit link will be displayed.
   - 2.2. Add “Offer A Training” link in next to edit link. This will be displayed only for the user of type “company” only, which will offer the current opened student user a training position by adding a record to “students_applications” table.
   - 2.3. A company user can only offer a training for any student for one time, if the company offered a training for a student, then the company should see “Offered” instead of the “Offer A Training” link, with the status of the application next to it (sent, approved, rejected).
   - 2.4. A list of company offers should be displayed under the student details ONLY if the current user is the student himself (the user id of the student). The list should include the

company name, the offer date and the application status. It should be ordered using the offer date.

   2.5. The student can accept or reject the company offer by clicking “Accept” or “Reject” link next to the offer listing, and this will change the application status based on the clicked link.

3. The “add-student.php” page (for adding and editing students):

   3.1. Only the user of type “student” and only if he has **no** student record, then he can add new student, and it will belong to him (the logged in user) when adding it to the database.

   3.2. Editing a student will not change the user associated with the student, only will change the student details.

4. The “index.php” page:

   4.1. The main page of the student user should display either a message: “You have no student details”, or a list of all of the applications (offers) for the current logged in student user.