2703ICT - Assignment 2

Due Date: 9am Monday 9 October 2023 (Week 12)
Weight: 50%
Individual Assignment

Introduction
In assignment 2, you will create a web application for allocating students to Work Integrated Learning (WIL) projects. WIL is a capstone course in IT/Computer Science degrees. The course sources projects from industry partners and assign student groups to work on the project.

The web application you are building allows industry partners to advertise project, and students to apply for projects they would like to work on, and then assign students to projects.

Details
Your implementation must use Laravel's migrations, seeders, models, ORM/Eloquent, route, controllers, validator, and view/blade. You have some freedom in designing your website, however, it must satisfy the following requirements:

1. Users need to **register** before they login to access the functionalities of the site. There are two types of users:
    a. Industry partner (InP),
    b. Student,

   Users need to select their type when they register.
2. All registered users can **login**. In fact, to be able to access any feature of this system a user must login. Once logged in, the username and the user type (InP or student) will be displayed at the top of every page.
3. A logged in user should be able to **log out**.
4. The home page shows a **list** of InP's names, clicking on an InP will bring up the **details page for that InP** which shows the InP name, their email address, and all the projects (title) offered by this industry partner. Clicking on a project will bring up the **details page for that project** which contains all the information about that project including the name of the InP that offered this project.
5. InPs, and only InPs, can **create/add project** to the system. There is a form where an InP can enter new a project. A project has a title, InP's name and email address, project description, and the number of students needed. Title and name must be more than 5 characters, email address must be an email address, description need to be at least 3 words, the number of students (team size) must be between 3 to 6. A project is added for a particular trimester in a particular year (called offering). The valid trimester is 1 to 3. If there is any validation error, the error(s) will be shown next to the form and the form will contain the previously entered values.
6. InPs can **update** and **delete** the projects they offer (but not other InPs projects). The update form must have existing data. With delete, if a project has students applied to work on that project (see requirement 9), then the project cannot be deleted. When a InP tries to delete a project with application, s/he will be redirected to the project's details page, and an error message will be shown to explain why the deletion is not possible.
7. Projects in the same offering **cannot have the same name**. Proper error message should be shown next to project create form when an InP tried to create a project to have same name as another project in the same trimester and year.
8. There is pagination in the **home page**, where only 5 InPs are displayed per page.
9. There is a **projects-list page** that displays all projects (names) **group by the year and trimester** of offering. For each group of projects, the grouping should be obvious and the year and trimester for that group should be clearly displayed. The groups are displayed in reverse

chronological order, i.e. the latest year/trimester should be displayed at the top, follow by the 2nd latest group. Clicking on a project will bring up the details page for that project.

10. A student can **apply** to work on up to 3 projects. When applying, the student needs to submit a short text justifying why they want to work on this project. As a part of this feature, the project details page will show all the students that have applied for this project and their justification.
11. When an InP offers (creates) a project, s/he can **upload images and pdf files** which contains more information about that project. These uploaded images/files will be accessible from the project details page for that project, such that:
    a. All images will be shown.
    b. There will be links to download each pdf file.
12. There is another type of user called **Teacher**. There is only one teacher that is set via the seeder. Teacher has the following abilities:
    a. Before a registered InP can create a project, s/he needs to be approved by the teacher.
    b. Teacher can see the profile information for all students (see requirement 13).
    c. Teacher is able to trigger automatic assignment of students to projects for a particular offering (see requirement 14).
13. This is an extension to requirement 10. Before a student can apply for any project, they need to provide **profile information** about themselves. The profile information student needs to provide are their Grade Point Average (a number between 0 to 7), and the roles they can take on. There must be at least the following roles for students to select from: software developer, project manager, business analyst, tester, and client liaison. A student must select at least one role and can select multiple roles. You are free to collect additional information to help you with requirement 14. Each student has a profile page that shows that student's profile information. This page can only be accessed by that student and the teacher. Student can update their profile information.
14. Once the teacher triggers the **auto assign** function, the system will automatically assign students to a project in the select offering (year and trimester). The scope of the assignment is to assign all students that have applied for one or more projects in that offering to a project in that offering. Furthermore, the conditions that the auto assignment should try to satisfy (in order of priority) when assigning students to a project group are:
    a. There is at least one student to take on each of the roles listed in requirement 13. It's likely (and expected) that a student may needs to take on multiple roles.
    b. Students assigned to the same project should have similar GPA.
    c. The team size is close to what the InP nominated.
    d. Students have nominated to work on that project.

    You are free to come up with more project assignment condition that can provide a better experience for students. Good conditions will be rewarded.

    It is possible for some projects to not be assigned any student. But all students that have applied for that offering should be assigned to a project.

    In reality, this is an over constrained problem. It is impossible to satisfy all the constraints. Hence, the expectation is for you to come up with a solution that you can demonstrate/explain how your solution can satisfy as much of the above condition as possible.

    The students assigned to a project will be displayed in the project details pages. In addition, the profile information of the assigned student will also be displayed (to make it easier to determine the quality of your auto assign feature).

Technical requirements

- Use Laravel's migration for database table creation.
- Use Laravel's seeder to insert default test data into the database. There should be enough initial data to thoroughly test the retrieval, update, and deletion functionalities you have implemented.
- Use Laravel's ORM/Eloquent to perform database operations. Only partial mark will be awarded for implementations using SQL or query builder.

- Proper input validation for all inputs. You must NOT implement any client side (html/Javascript) validation, so your server-side validation can be tested.
- Proper security measures must be implemented.
- Good coding practice is expected. This includes:
    - Naming: using consistent, readable, and descriptive names for files, functions, variables etc.
    - Readability: correct indenting/spacing of code.
    - Commenting: there should at least be a short description for each function.
    - View: proper use of template and template inheritance.

Documentation
Provide the following documentation in no more than 1 page:
1. An ER diagram for the database. Note: many-to-many relationships must be broken down into one-to-many.
2. Describe what you were able to complete, what you were not able to complete.
3. Reflect on the process you have applied to develop your solution (e.g. how did you get started, did you do any planning, how often do you test your code, how did you solve the problems you come across).

If you have completed the auto assign (requirement 14), describe how you have implemented this feature. Be convincing regarding how your solution satisfies as many constrains as possible. You can write up to a page for this section.

Reference and the Use of AI
If you have used code snippets from the Internet, you need to reference your source.

In this course we do not discourage the use of AI/Chat Bot. You are free to use AI to help with your assignment. The referencing requirement for the use of AI is that you copy and paste (or screen shot) your question/prompt and AI's answer into the reference section of your submission. Or provide a link to the chat log (by using the Share Chat feature of Chat GPT).

For further details of the requirements, refer to the marking rubric. **All requirements from both the assignment specification and marking rubric must be satisfied.**

Submission Requirements
You must submit the following items for the assignment:
- A compressed file containing ALL source files in your submission (including all PHP code), but **excludes the vendor and node_modules directories**.
    - This file must be submitted via the LMS through the assignment 2 link.
      Note: Delete the *vendor* and *node_modules* directories before you compress the files. (Restore the vendor directory before your demonstration with the command: *composer update*. The node_modules directory can be restored by the command: *npm install*). Use the *zip* command to compress your assignment directory (see Lecture 1-3).
- A PDF file containing your documentations.

Note: You are responsible for regularly backing up your work. Hence, if you lose your file due to not backing up, then expect to be heavily penalised.

Assignment Demonstration and Marking
After you have completed your peer review, you must demonstrate and explain your work to your tutor in Week 12 lab to have your submission marked by your tutor.

If you do not *demonstrate your assignment* to your tutor, your submission will be regarded as incomplete, hence you will not receive a mark for this assessment item!

During the demonstration, you need to show the last modified date of your file (on Elf, run the command: *ls -la* in your *routes* directory).

Warning: We take student academic misconduct very seriously!