Computer Science Department
Web Application and Technologies (COMP 334)

Due Date 09/02/2023, see submission instructions below

COMP334: Project Description

IMPORTANT NOTES

- You should not in any case use any external web development tool. Any try to use such a development tool will get you a mark of **zero**.
- Submission is only through your domain that we created for you.
- Since you are submitting to your domain at CSHost, it is your responsibility to make sure that your scripts such as file upload, database schema, etc will work properly on CSHost server. Make sure that you upload your scripts and other development files before submission date so that you have some time to test and fix any pop-up issues.
- You must define your data base connection details in a separate file called "db.php", it should **define** the required variables used to create a connection, and ***create a PDO object***.
- You should always use **ONLY** prepared statements with named binding parameters for any database SQL.

INTRODUCTION

Online team management applications are essential web tools that help in managing and coordinating projects and daily tasks among team members. Through these applications, tasks are available online and thus, can be accessed, shared, and collaborated from anywhere at work, at home, at school, etc.

You are required to create a dynamic web application for team management using HTML, CSS, PHP, and MySQL database. The main goal of this web application is to enable team members to assign tasks to and track tasks with other team members. Your application should be simple to use with consistent look and feel among all web pages.

SYSTEM MAIN FEATURES

Your web application should provide the following main functional features:

1. Team member registration: before team members can join your team, they should register and create their account. After creating their account, they can login and start using the system, i.e., assign, view, update tasks, etc. This also means that only **authenticated** members can use your web application.
2. Task management: through this feature, team members should be able to:
    a. Assign new tasks to themselves or other members. The same task cannot be assigned to more than one member.
    b. View their own tasks and update them, for example changing the **status** of a task (pending→ active→ finished)
    c. When team members login to web site, the system displays their daily tasks. However, they can also search for tasks according to date, status, and assigned members.
    d. Since those tasks will be displayed in a list, every task will have a different background color depending on its status.
    e. A team member can view (read only) other team members tasks.
    f. Since that tasks lists will be displayed in tables, you should provide the functionality so that tasks can be **sorted** according to date, priority and status.

TEAM MEMBER INFORMATION

The following is the information that the system should record for each team member when registering:

Note all member information are **required.**

Member Registration done in two steps:

a. Getting member details: name, Identity Number, Nationality, Address, Mobile Number, email address, photo, qualification {education, training}, working experience, upload CV.
b. When the member information sent to the server, the system asks the user to create an eAccount by selecting a user name and password. Upon successful creation of the e-Account the system must generate a 6 digits member ID, user name and password are used as user credentials to authenticate on the system.

TASK INFORMATION

For each task, the system should record the following information:

1. Task title, **required**
2. Task description: **optional**
3. Start date: **required**
4. Due/end date: **required**
5. Priority 1-3, 1= low and 3 = high priority, default is 1. **optional**
6. Assigned-to member: the member that the task is assigned to, **required.**
7. Assigning member: the name of member who assigned the tasks

DETAILED SYSTEM REQUIREMENTS

- When member logs into web site, he/she are presented with their daily tasks, that is, the tasks that are due on that date.
- Tasks' status can be explained as follows:
    - When tasks is first assigned the system sets it as **pending.**
    - When user starts working on a task, he/she should change its status into **Active.**
    - When user finishes his/her task, he/she changes its status into **Finished.**
    - If task passes its due date and still not finished, the system automatically sets it to **Late.**
- The system should provide a special search page that enables members to search and view tasks according to:
    - Dates
    - Priority
    - Status: view late, active, finished or pending tasks.
    - Member name: view tasks assigned to another members.
        - View late or active etc tasks of other members
- The system should display the profile photo (in small size of course) next to each task for the member who assigned the task.
- All application pages should be authenticated (only logged users can access them)
- Your page design should be consistent, containing header, navigation and footer sections. However, these sections are dynamically generated by your application and **not static.**
- The first page to load when user first access your web site is the login page.
- When user makes successful login, the system displays the main page with daily tasks.

- All pages contains:
    - Navigation section on the left which contains, Figure 1 below provide an overview of the page layout:
        - Home page link: this returns the user to the main page which displays the daily tasks.
        - Search page link.
        - Add new task link.
        - Late tasks link: view late tasks
        - Pending tasks link: view pending tasks
        - Active tasks link: view active tasks.
    - Header section containing:
        - Your logo
        - A name for your web application
        - About us page link.
        - Contact us page link
        - And log out link.
    - Footer section: smaller logo and copy write text, such as address, contact email and telephone number for customer support.
    - Main Content section: this section differs from one page to another, i.e., all pages have same header, navigation and footer sections and differ in the content section. By default when user first login, the content section displays a list of daily tasks, i.e, the tasks for that date.
- The tasks list, which is a table, should contain task title, the photo of the sender team member, the photo of the receiver team member, priority and special background color for its status.
- The task list should also contain a link to open full task details in another page and enable user to change task data such as its status.
- In the user forms, all fields that are required should have special background color. User CSS classes for this option.
- System should mark input field with special background color if user forgets to enter required field. System should also display appropriate error message near that error input field.

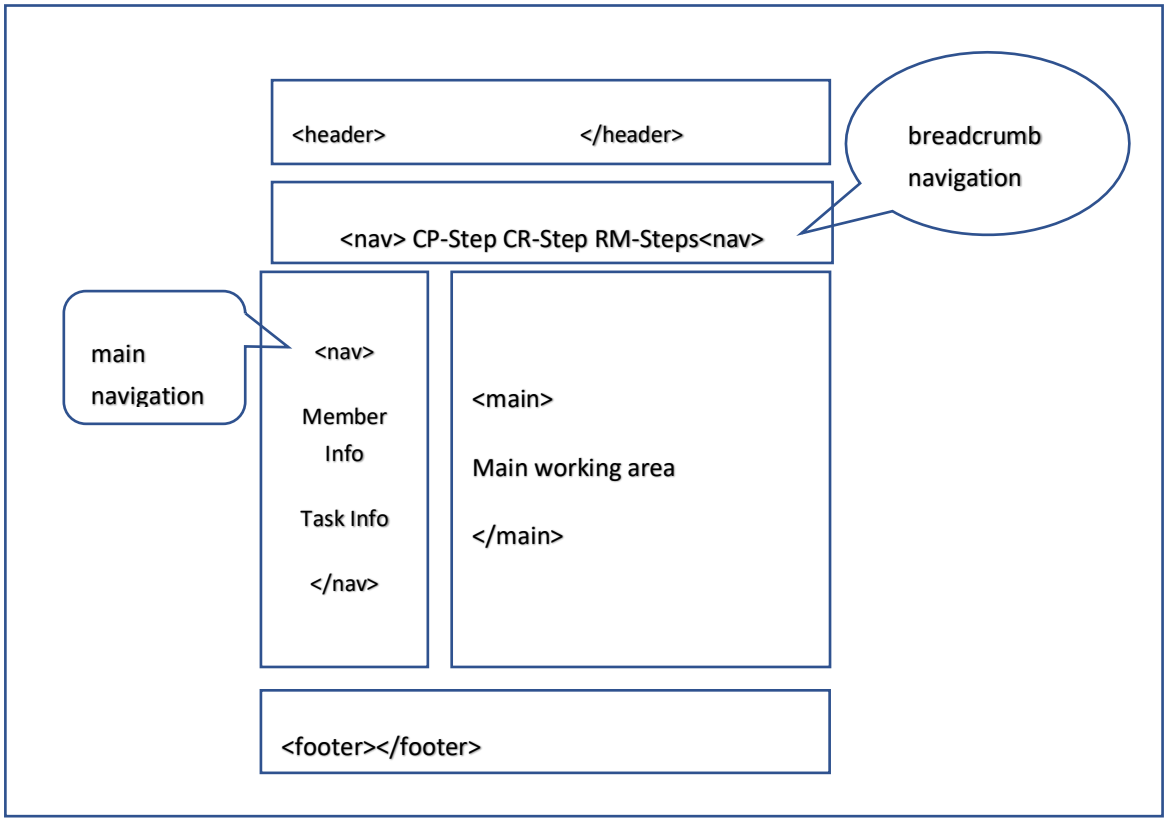

Figure 1: Page layout template

Note: A breadcrumb navigation provide links back to each previous page the user navigated through, and shows the user's current location in a website. You should provide different style for each steps, for instance different background or/and text and for style: (Completed Step: CP-Step, Current Step: CR-Step, Remaining Step: RM-Step)

CSS REQUIREMENTS

- Use only external CSS files, NO embedded or inline CSS are allowed.
- Make sure you use the CSS rules that are given to you in lecture ONLY!
- It must use flex attribute to implement the side-by-side layouts
- Use the correct element types, for instance nav for navigation bar, etc.
- You must use classes and ids in your solution, in a right way, for example: create a class called “Completed” to give a style for the completed tasks.
- You must use semantic elements, general elements such as div is allowed unless if there is no available sematic element, use instead of <i>, instead of <b> and provide style

- You should provide a style for the **required** fields and must be consistent with all forms. All mandatory inputs should be clearly marked, *even if all the input fields in a form are required they must be clear for the user*.
- Change default font properties, in 4 locations, headers, search result table, .., etc
- Change box model properties (border, padding, margin), in four different locations in your website
- You must use figure element to display the images, and change the position properties to display the images and the captions.
- *You must submit the final styles rules, all the classes must be used within the HTML pages. All CSS file must cleaned form all temporary and not in use classes.*
- *Any work taken from the internet must be documented, by adding refence to the source. The code {HTML, CSS, and PHP} taken form the internet or somewhere else should not exceed more than 10% of your total work, and* ***it should not be a complete functionality****.*
- All input text should change background on focus
- All input labels should be surrounded by a border in a different color than text label color.
- The table header should have a different background and text color than the table data. Headers data should be centralized.
- Table result should have a border and consume 100% width of the containing element.
- The task reference (that appear in search table) should given a different style than the other links within the document.
- The table rows should have a different background for even rows than odds rows.
- Links outside your website should have a different color schema (color for visited, non visited, and hovering) than those within your website. The difference should be clear enough to be disguised. Also, they should be different on text decoration.
- Clickable image should display a border when hovering.
- Lists: unordered list select an image to use it. When unordered list appears appeared within another unordered list use a different image. Ordered list use Arabic digits for a top level, and alphabetical for the nested level.

BONUS REQUIREMENTS:

You will get bonus marks for doing the following:

- In the navigation section, add a number next to “pending”, “active”, “late”, etc to indicate to user how many tasks are in that category.
- For each table displaying a list of records in your web application, the number of displayed records = 10, and user can view the other record by choosing page2, page3, etc feature at end of each table.

SUBMISSION REQUIREMENTS:

- The Deadline for submitting this task is 09/02/2023. ***At 16:00 at the CShost***, so the upload ftp service will be closed on 09/02/2023 at 17:00. *In addition to your submission to the CShost you have to send me your project to ITC the deadline is* 09/02/2023*.*
- Project Discussion will start on Sat 11/02/2023, see discussion instructions below.
- You must upload your folders and files, including the SQL and insert statements, to your web hosting server inside a subfolder called “project” inside the root web/ folder, and update you home page to include a reference link to you project index 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 ITC 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.