Term Project – Phase 2 Description

Using the pages you created in the previous phase, we would like you to add the back-end functionality of your project using PHP + database. Back-end in this project involves creating/populating the database, adding functional log-in, log-out and sign-up functionalities (using Sessions) and the ability to:

- **Employee**: view their list of requests, add a new request, and edit a request.
- **Manager**: browse all requests according to the service type, approve a request or decline it
- View the request information page for a selected request.

In order to implement these functionalities, you need a database that contains tables for employees, manager and request.

In order to fulfil these requirements, please develop the following:

1. **Create a database** following this schema (underlined is PK, dotted underlined is FK):
   ```
   Employee (id, emp_number, first_name, last_name, job_title, password)
   Manager (id, first_name, last_name, username, password)
   Request (id, emp_id, service_id, description, attachment1, attachment2,
   status)
   Service (id, type)
   ```
   Where:
   - attachment1 and attachment2 are links to the actual files that are stored in the system.
   - empNumber referenced to employee's identification number

   You can add more data if you have extra needs in your application.
2. Add at least three rows per table except manager only 1 row.
3. Add log-in functionality to the manager and employees log-in pages. When filling the log-in form, your form should call a PHP page that checks if `username/empNumber` and password are correct, if yes, the user's id and their role (i.e., Manager or Employee) are added as **session variables**; then the user is redirected to the appropriate home page. Otherwise, they are redirected back to the log-in page with an error message.
   Please note: passwords are never stored as plain text into the database, hash passwords before storing them in the database using the PHP function password_hash.
4. Add **security** to the rest of your project pages such that all pages check that the user has logged-in, if not, redirect the user to the home page (hint: use session variables).
5. The **sign-up page** has a form that, when submitted, calls a PHP page that will add the new employee to the database then redirect to the employee home page if they signed up correctly. If the employee's identification number is already in the database, redirect the user to the sign-up page with an appropriate message.
6. The **employee's home page** has a button that directs to the 'add new request' page and a button that logs the user out. Additionally, the employee's home page has PHP code that:
   a. Checks the employee's id (from the session variable), retrieves their information (first name, last name, employee's identification number and job title) and displays them in the page.
   b. Retrieves all the request in the requests table that belong to the employee and prints two lists of these requests one presents in progress requests and the other for processed ones.
   c. For each request in the list, it is displayed as a combination of request's ID and service type. Also, the code generates a link for viewing the '**Request information page**'.
   d. Display beside each request a link that enables the employee to **edit** the request which links to the **'edit request page'** which is a form that should be pre-filled with current request information using PHP and allows the employee to edit the information and submit changes. When submitted, it calls a PHP page that will update the request information in the database, and then redirect the user to the request information page with an appropriate message.
   e. Display the status for processed requests.

7. **‘Add new request’** page has a form that takes the request information, when the form is submitted, it calls a PHP page that adds the request information to the database, then redirects the user to the new request information page.
8. The ‘**request information page**’ has now PHP that:
   a. Checks the id that is sent in the query string, retrieves the request information from the request table and displays them appropriately in the page. It also displays the employee’s full name and the service type.
   b. The attachments are displayed if they were images. otherwise, they are appeared as links.
   c. If an employee is viewing the request page, an ‘edit’ button is displayed that redirects to the **‘edit request page’**.
   d. If a manager is viewing the request page and the request status is under processing, ‘Approve’ and ‘Decline’ buttons are displayed. If the status of the request is approved only Decline button appears while in declined requests only Approve button appears. When the user clicks ‘Approve’ or’ Decline’ buttons, a PHP page will update the status of the request in the database and refresh ‘**request information page**’.
9. The **‘Manager home page’** accesses the current manager’s id from the session variable, retrieves their information from the database and displays them appropriately in the page. The page also gives the manager the ability to ‘sign-out.’ The page also checks all the requests that are existed and categorised them in different lists according to their type.
   In each list:
   a. The Request’s name that consists of request ID and employee’s name can be clicked. It directs the user to the request information page.
   b. The list of requests is sorted so that the requests in which they are under processing appear at the top (consider also having a different style, colour, etc. for under processing request)
   c. Each under processing request will have next to it two links that gives the option to ‘approve’ or ‘decline’ the request the links should do as the ‘approve’ and ‘decline’ buttons in ‘**request information page**’ but it redirects the user back to the manager home page.
   d. Each approved request will have next to it a link that gives the option to change the status to declined.
   e. Each declined request will have next to it a link that gives the option to change the status to approved.

10. A sign-out page that for both manager and employees that are linked in their home pages. The page should wipe out the session and then redirects the users to the home page.

You need to submit your NetBeans project folder after exporting it. It should include all your website pages (HTML, CSS, JS, images, and PHP) in a working hierarchy. In addition, you also need to submit the exported DB. One member of each group should submit the phase files via LMS. An LMS link for your submissions will be announced soon.

Submission deadline: Before 11:59 pm on Sunday 27 March.