Job description

2.1 and 2.2

During the project, I used HTML, CSS, PHP, MYSQLI, Javascript, and Bootstrap. To run the program, you need to have XAMPP (MariaDB database and Apache HTTP server) installed. You need to add the "primeholding" folder to the "htdocs" folder inside the XAMPP folder. After starting the server, the program can be accessed through the browser using the address http://localhost/primeholding/index.php. If you don't do this, you won't be able to run the program.  
  
In the folder is download database which name is workforce.sql, you need to import it to database so you can see information. Without that is not possible to see anything.

First, I created the HTML structure and then styled it using CSS. After that, I created a database (MySQL) where the "employee," "admin," and "tasks" tables are stored. The entire database is linked to PHP codes, where data can be manipulated directly through inputs, which is practical.

The "employee" table has columns for ID, full name, phone number, date of birth, salary, and user\_type. I added the "user\_type" option because I planned to have one login page for both employees and admin, but I didn't include a password for employees. Therefore, I had to have separate logins for the admin and employees.

The admin has a username, email, and password. There is no option on the index.php page to add another admin, so it must be done directly in the database. On the index.php page, there is an option to create an account where a person who wants to work needs to register. After registration, the data is saved in the database, and the person can log in through index.php.

Once logged in, the person will see a table with a title, description, assignee ID (each employee has their own ID), the deadline for the task, and an option to mark it as completed. If the option is set to "1," it means the task is completed. If it is set to "0," the person can click on the "yes" button after completing the task to change it to "1."

After clicking "yes," the information is forwarded to the "tasks" table, which is managed by the admin. The "is\_completed" column in the table is set to "0" if the task is not completed and "1" if it is completed.

Admin options: The admin's username is "Admin," and the password is "123." Once the admin logs in, they will see CRUD options, where they can read, add, modify, and delete data. Additionally, there is an option for tasks, where the admin can perform CRUD operations. When assigning a task to an employee, the assignee ID is entered, which is directly linked to the employee's ID. I linked them directly to avoid typing the employee's full name every time a task is assigned.

When a task is completed and the "yes" button is clicked, the "is\_completed" column in the table is changed to "1." The "total\_complete\_task" is then calculated based on the "assigneeId" linked to the employee's ID with "is\_completed" set to "true." This way, the completed tasks can be viewed. To view completed tasks, access the address http://localhost/primeholding/tasks.php, where there is a button for "efficiency," which shows the order of employees who completed the most tasks.

2.3 I used PHP as the primary programming language because I could create a database and manipulate data easily. I have the most experience with PHP, although I used Javascript in some situations. MariaDB server is more natural and easier for me to use than others, which is another reason why I chose to use PHP.