QUANTOX

SOFTWARE DEVELOPMENT & IT CONSULTING

PHP Backend Internship Test

THE GOAL

The goal of this test is to assert (to some degree) your coding skills.

Feel free to add at any point any particular technique or algorithm that you think might enrich the overall quality of the end result.

All the stuff you do needs to be tracked on a Git repository, so we can see your thinking.

REQUIREMENTS

PHP

You **MUST NOT** use any known frameworks for this test, but to create one yourself.
You are, however, allowed to use any PHP libraries (also other frameworks libraries) as long as they are installed via [Composer] (https://getcomposer.org/).

DATABASE

The database engine used MUST be MySQL.

EXPECTATIONS

The candidate is expected to be able to develop this test in 7 days, although feel free to invest as much time as you find adequate. If you find any of the requirements unfamiliar, take your time, do the research and try to implement it.

We would like to see:

- simple MVC implementation
- full OOP and Namespace use
- CRUD operations
- Use of some design patterns and SOLID principles
- Regular git commits with understandable comments

YOUR TASK

Please implement the following 4-screen logic:

HOME SCREEN

- Link to login screen
- Link to register screen
- Search form:
    - Search text input
    - User type select field (Front End Developer, Back End Developer)
    - Search button

Description

- The link to the login screen should send the user to the Login Screen.
- The link to the register screen should send the user to the Register Screen.
- By typing any text in the search field and submitting the form, the app should redirect the

user to the Results Screen and display the list of matching results.

- Every type of user could have subtype(s):

- Front End Developer
    - Angular
        - AngularJs
        - Angular 2
    - React
        - React native
    - Vue
- Back End Developer
    - PHP
        - Symfony
            - Silex
        - Laravel
            - Lumen
    - NodeJs
        - Express
        - NestJS

REGISTER SCREEN

Display a form to the user with:

- user type select field (user can select any of type or subtype)
- email input field
- name input field
- password input field
- repeat password input field
- submit button

When the form is submitted, it should be validated and, if the validation passes, the information should be saved to the database.
If the validation does not pass, an error message should be sent back to the client with a description of what went wrong.

LOGIN SCREEN

Display a form to the user with:

- email input field
- password input field
- submit button

When the form is submitted, it should be validated and, if the validation passes, try to login the user.
If the user is not logged in successfully, a generic error message should be returned to the client, like “Error logging you in”.
If the user is logged in successfully a message should appear saying "Welcome, {name of the user}!" on the dashboard page.

RESULTS SCREEN

- If the user is logged in, then the page should display all matching results of users with either a name or an email address similar to the query text provided by the client.
- If the user is not logged in, no results should be displayed. Instead, query text should be remembered and a message saying “Please login” should show, with the login form below (the same "view" file used on the Login Screen should be used here).
- After login or registration process (and query text is remembered) user should be redirected to the results screen and there are displayed all results filtered by remembered query text.
- Results Screen should have two sections:

    - **Left side** - In this section, you need to show the list of all User types with a count of users in the type (parent type also should count all users from children types)

      Example:

      Back End Developer (5)

        PHP (3)

        ....

        NodeJS (2)

...

We need to count only users from results.

Bonus: It will be great if you make and show the list expendable (for example like directory tree)

- On the **right side** - we need to show users from results.

TEST EVALUATION CRITERIA

- Front End look and feel will be valued, but will be considered as a plus.
- All validations must be server-side. You can use javascript, but that’s completely optional.
- Installation instructions should be included.
- The code should be documented where needed (use your own judgment).
- Although not mandatory, you can include a text file explaining your approach to the problem and why it was resolved in a certain way.
- Any extra features that you might want to include will be taken into account AS LONG as the basic feature requirements described in this sheet are met.
- The security of the overall solution will be taken into account.
- Design patterns **and other coding best-practices will be taken into account.**

THANKS, AND GOOD LUCK!