Code Test at Crystal D

Thank you, Toua and Mark, for providing me with the opportunity to attend the Code Test for Crystal D. It was enjoyable and a great learning experience for me incorporating PHP, MySQL and Vanilla JavaScript. I would like to walk you through my experience about working on the problem and it requirements.

Thought Process and the Journey behind Solving the Problem:
I am going to break down my thought process of solving the assignment and share with you step by step what I have done.

After following the rule of "Divide and Conquer", the application roughly comprised of the following parts:

I. Connecting with a server:

I am using XAMPP server and start the Apache server from XAMPP Control Panel

II. Connecting with MySQL database and creating user credentials and privileges for that user:

I opened the command prompt in my MySQL server bin folder. Next I connected the MySQL server with the XAMPP server using script in command shell. Then I accessed my database by creating blank database, specific user, password and granting all privileges to the user on the database

II. Writing the SQL query to retrieve information about people and their hobbies:

I created all the tables in the blank database from codetest.sql using SQL query import in MySQL Workbench

III. Connecting to the MySQL database using PHP with appropriate user credentials

IV. Creating the stylesheet in index.html to show the table for people and their hobbies according to the codetest_ui.png (all rows of data and buttons):

index.html contains the style for table, button, header click, button and button click.

V. Using ajax GET to fetch information from the PHP code with SQL query about people and their Hobbies to the JavaScript part to show the information about people and respective hobby:

createTableRow(person) function creates row entries from the SQL query with ajax GET request from the PHP endpoint to JavaScript

VI. Sorting Tables on Header Click with JavaScript sortTable Function (in script.js):

The sortTable(columnIndex, ascending) sorts entries depending on any given column and shows down caret sign (index.html: th.desc::before) when the row values are in ascending order of the given column

VII. Using ajax POST request to handle button onclick for changing hobbies in each row:

Here I have called ajax post request from script.js and handled the POST in PHP where the request sends the id of the person whose hobby needs to be changed to PHP and based on the id a new hobby is randomly chosen by SQL. After that the data retrieved from SQL is sent over to JavaScript endpoint

VIII. Handling SQL Injection attack:

I have particularly used PDO for two reasons. Firstly it supports different databases and secondly according to many users online, using PDO to connect MySQL can protect over basic SQL Injection attacks (SQLI). The SQLI which happen over dynamic parameters in mysqli and need to be explicitly handled with escape string can be avoided using PDO. The PDO follows special template to handle dynamic parameters. PDO covers the data sanitization with named parameters, question mark and through binding the parameters through their values.

❖ Special Attention was given to:
    o Reusable Functions: the JavaScript functions createTableRow(person) and sortTable(columnIndex, ascending) and change_hobby(id) have

been defined as functions rather than inline using vanilla JavaScript as they are repeatedly called within the lifetime of the Application

Although the functions in PHP like database connect and database disconnect could be written as functions, it was avoided as they happen only once while the application runs. Also get and post ajax requests are controlled from JavaScript, their response in PHP were written as inline code.

- Separation of HTML, JavaScript and PHP: It was one of my greatest concerns to organize and write clean segments of code according to the test standard. I have written CSS and HTML portion in index.html, PHP code in api.php, and JavaScript functions as well as ajax get and post requests in script.js. Running the index.html will fairly demonstrate my work.

❖ Testing:
I have tested all the SQL queries on MySQL Workbench First and then incorporated in the PHP. I have also tested JavaScript functions and the CSS, HTML on dummy data when I wrote the functions and style and body scripts initially. Header Click function has been checked on all different columns.

❖ Real Product Specification vs Ideal:
The final applet matches almost all the requirements. I was skeptical to send the person id with "GET" request, hence I used "POST" request. Unfortunately, the ajax post request on clicking the "Change Button" did not operate. I would be happy to troubleshoot and discuss the matter with you. I have written all the code for POST request in JavaScript, PHP, and MySQL. To test, I printed out the person id in JavaScript Console which works, and the SQL query works on the MySQL Workbench separately.

I am attaching a video link to show how the applet looks from my side. (https://drive.google.com/file/d/1XazCBEW7UJ1MAVEt6q77TmlpOvHhIOrh/view?usp=share_link)

My github repository for the project: https://github.com/Sejuti048/code_test-main

My overall experience has been great working on this project. As I mentioned earlier, I love learning new technology and implementing simultaneously. It was great to catch up with all the new changes in PHP and MySQL after 2015 when I built a Software as my Junior year project. Thank you Mark and Toua again for giving me this opportunity. It will be an honor for me if I can go to the next step and join the amazing team.