The University of Technology, Jamaica

School of Computing and Information Technology

Web Programming (CIT2011) Sem1 AY22-23 Group Project

Due: Week of November 21st 2022 **Groups of 3 or 4**

Your assignment must be placed in a zipped folder and uploaded to a location determined by your tutor. Any assignment submitted after due date/time will attract a 10% deduction for each day up to the third day. Assignments submitted after three (3) days will attract a zero (0) grade. Give your group a team name.

Basic requirements:

a) Creativity - Presentation and Feel of the finished product, eg color, font, readability, design/layout of web pages.
b) Completeness of the website (working hyperlinks, seeing images, appropriate tags, headings, documentation of code, etc)
c) Use and style of HTML5 tags.
d) Content (including complying with the requirements)
e) Ensure that all form field tags have a unique id attribute for JavaScript to access them.
f) A copy of the XHTML Validation report.
g) An agreed upon workplan indicating each group member's responsibilities, the name of JS functions and user defined functions used including a short description, link to hosting, storyboard including wireframes, sitemap.
h) A 10-15 minutes pre-recorded demo of your completed website. Each member of the group must speak.

Instructions: Perform all the requested tasks given below.

Description: You are required to build a *Game similar to Jeopardy*, which mirrors the Jeopardy game. In this game there are a maximum of three players, but the user may choose the individual option or Multiple Players Option (**Multiple Players option will not be enabled**)

You are required …

For this game the user will be playing solo. At the start of the game, the player's total is $100

The user interface must be organized into 3 sections: (1) Registration, (2) Play, and (3) Results areas.

Task 1: (6 marks)

Create a new HTML page called *index.html*. There is to be a form on the web page with the following fields for entering information on each player: *Player's First and Last Name, DOB,* Email, Age, *Gender, Address (street, city, town, country)*, educational level and image of self.

i) Player's name, DOB, Gender, town and educational level are required values.
ii) Age must be calculated by the system from date of birth and must be greater than 12.
iii) First names must be more than three (3) characters in length.
iv) Email address must be validated to only accept email addresses that end with "@SomeEmail.com".

Task 2: (7 marks)

Create a JavaScript function called **Register()** that will accept the values entered in the fields listed in Task 1, validate all entries using JavaScript validations, and then append the validated content to a global JavaScript array called **PlayersData[]**.

Task 3: (4 marks)

Add a button, called *Register* to the entry form created in *Task 1* and ensure that this button calls the function, **Register()** from *Task 2* whenever it is clicked. After the Register button is clicked, all the fields from *Task 1* above and *Register* button must be disabled, additionally an **End Game** button along with the **Play** button (from Task 5, below) must now be enabled.

Task 4: (20 marks) - Requirements

Create a JavaScript function called **PlayGame()** that will:

a) Call a function to display the Jeopardy board. The Jeopardy board has five different categories of questions, and each category has 5 levels of questions. The board would look something like below (You may determine your own categories):

Category 1	Category 2	Category 3	Category 4	Category 5
$100	$100	$100	$100	$100
$200	$200	$200	$200	$200
$300	$300	$300	$300	$300
$400	$400	$400	$400	$400
$500	$500	$500	$500	$500

b) Allow the user to choose a topic category and a question in that category associated with a specific dollar amount ($100, $200, $300, etc.), e.g., '*Vertebrates for $300.*'
c) The question in the topic category is displayed.
d) There should be a 60 second period for answering a question.

e) The Timer begins to count down.
f) The player will then answer the question:
    a) If the Player answers the question correctly withing the 60 seconds period, then add the value amount of the question to the players current score.

    b) If the player doesn't answer the question in that 60 second period then
        i. they forfeit the question
        ii. the question goes back to the board
        iii. the player loses that dollar amount from their total amount
        iv. Player may choose an option to reveal the answer before moving on. If the answer is revealed then that question is disabled on the board.

g) A "**Daily Double"** should be hidden behind at least three (3) randomly chosen $ figures on the board. If a user selects that question and it is a daily double then:
    a) they can choose a dollar amount from $0 to the total amount of money they currently have.
    b) If the answer is correct, they get the amount they wager (they still get the same 60 second to answer).
    c) If they are incorrect, they lose the amount they wager
    (Since the Daily Double is randomly placed, it should be placed at a different location each time the game is played)

h) Final Jeopardy
    a) Upon successfully completing Level 1 of Jeopardy the user has the option of playing Final Jeopardy.

    ♣ A category is announced by the computer randomly. The player will enter a wager, based on the category, of as little as $0 or up to as much money as they have accumulated.
    ♣ The contestants have 60 seconds to enter a response, phrased in the form of a question.
    ♣ If the user is correct, then the amount wagered is added to their total otherwise, it is subtracted.

i) Termination of the Game:
    a) Player may choose to Exit the game before it ends; or
    b) The game terminates when the entire board is disabled/all the options are selected.

Please Note*: Initially, Player's Total is a global variable and should be set to $100.*

Task 5: (3 marks)

Add another button, called *Play* to the entry form created in Task 1 and ensure that this button calls **PlayGame()** from *Task 4* whenever it is clicked. When clicked, this button also enables the Play area along with the Player's answer input, **Answer** and **End Game** button.

Task 6: (8 marks)

Create a JavaScript function called **CheckAnswer()** that will check and validate the answer provided by the player.

a) If the player answers the question before the time expires then
    i. If the answer is correct
        1. The question is removed from the board (that option is disabled or disappears)
        2. The user gains that dollar amount to their total
        3. Player can return to the Board for another chance.
    ii. If the answer is incorrect
        1. The user loses that dollar amount from their total
        2. Ask the user if they wish to know the answer.
            - If the user says yes then the answer is revealed and that question is removed from the board (disabled/disappear).
            - If the user says no then the question remains on the board.
        3. Player can return to the Board for another chance.

b) Appended to the global JavaScript array called **PlayerData[]: - the** player's total games played so far, - an entry to indicate whether the question was answered correctly - another entry for whether questions was answered incorrect, - and the player's current total.

Task 7: (2 marks)

Add another button, called ***Answer***, to the play area created in Task 5 and ensure that this button calls **CheckAnswer()** from **Task 6** above.

Task 8: (2 marks)

Add another button, called **PlayAgain** to the play area in **Task 5** and ensure that this button calls **PlayGame()** from Task 4 above.

Task 9 (3 marks)

Add another text-area to the form created in Task 1. Ensure that this textarea has the id attribute: '**showpercentage'**. This text-area will list each player's name along with their current percentage score, number of correct answers, number of incorrect answers and total questions answered separated by commas in a new line.

Task 10: (7 marks)

Create a JavaScript function called **findPercentageScore()** that will calculate and display total number of questions, the number of correct answers, the number of incorrect answers, the percentage score, and the player's name, town, current date, in the '**showpercentage'** textarea. You must always clear "**showpercentage'**" textarea before displaying all data in it.

Task 11: (3 marks)

Add another button, called *Quit* to the entry form created in Task 1. When clicked, the End button also calls the **findPercentageScore().** Clear the form from Task 1 and enabled inputs, all buttons must be disabled except the Register button. Play and Results area are disabled.

Task 12. (2 marks)

Add a new textarea to the form created in Task 1. Ensure that this new textarea has the id attribute: **'showallplayers'**. This is because it will be used to list all entered persons.

Task 13: (6 marks)

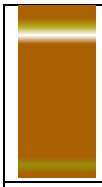
Create a JavaScript function called **showAll()** that will display all the data within the global array in the *'showallplayers'* textarea created in Task 11. Note, display should be properly formatted by showing each userName, Age, Number of Correct Answers, Number of incorrect answers, Final Total along with the percentage score, separated by commas in a new line. Show all must always clear "showallplayers" textarea before displaying all data in it.

Task 14 (2 marks)

Call **showall()** at the end of the **checkAnswer()** function ensuring that the data list in the global array, **PlayerData[]** is always updated when a new answer is provided.

Task 15 (10 Marks)

Note that JavaScript can change the body of a container tag like or by using the innerHTML property like: document.getElementById(*id*).innerHTML = <*string*> where string can be any set of HTML tags and content you want to see in that container and your container must have an id attribute for JavaScript to identify and change it. Also note that a small, pixelated JPEG or PNG image can be stretched horizontally to any size without losing its consistency in HTML5 using img's width attribute as can be seen below.

	
`<img src="./thinbar.jpg" width="20px"/>`	`<img src="./thinbar.jpg" width="100px"/>`

a) With this knowledge, you are to create a function called **showfreq()** which will display two frequency bar charts based on the demographics of all persons added to the system. Your first chart is to show a frequency chart on gender (how many males to females are in the system). Your second chart is to show the frequency chart on age (<20, 20 to 39, 40 to 69, >69).

b) For each chart, you are to count each instance of an attribute first. That is, count the number of females in the array; or count the number of ages between percentage score between 50 and 59 inclusive. You are to then use the total number of persons in the array to calculate these counts as a percentage of the total. That is: if 15 persons are females and there are 30 persons in the array, then the percentage of females is 50.0%. After you have all of the percentages for a chart, create the bar chart using HTML5 tables and stretched images (whereby the width of the image is the percentage value, example: females*: <img src = "thin_bar.jpeg" width = "50"*> noting that the % sign is left off.

c) The **showfreq()** function is be called and refreshed every 5 seconds once the page loads and it should display the two charts in a tag.

Task 16: (2 marks)

Add a new button **Results** to the webpage created in **Task 1** and ensure that this button calls **findPercentageScore()** from **Task 10** whenever it is clicked.

Task 17: (8 marks)

Creativity - Presentation and Feel of the finished product, eg color, font, readability, design/layout of web pages. Vanilla CSS and/or Bootstrap can be used; please indicate the type used.

Task 18. (5 marks) ***no project will be graded without the following submissions.

- An agreed upon WP Group Project Plan for each group member. Use the pdf template with the same name on Moodle.
- Include a signed Authorization form from each student.
- Word document with XHTML validation report – no errors, less than 6 warning
- An agreed upon workplan indicating each group member's dev responsibilities, the name of Js functions and user defined functions used including a short description.
- Link to hosting page/site.
- Wireframes of the website.