HTML, CSS, JavaScript – Assignment

This assignment is the combination of your learnings in the first half of the course. You build a Travel Agency website and add CSS, dynamic HTML and JavaScript features.

Value

The cumulative lab work you have been doing in this course is worth 30 marks.

Due

This assignment is due on the date specified by your instructor.

Directions

HTML

1. Create 3 pages named "index.html", "register.html", and "contact.html".
2. **<u>In the index.html page:</u>**
    a. Display an image at the top of the page that will be your travel agency logo.
    b. Next, display an "h1" heading with the text "Welcome to Travel Experts" inside it.
    c. As a challenge, see if you can find a way to make both of these display on the same line.
    d. On the next line, display some travel pictures (find any free images online).
    e. On the next line, display two images that are links to the "contact.html" and "register.html" pages (wrap them inside an anchor tag). Find images online that are relevant to these links.
    f. At the bottom of the index.html page display a copyright message with a copyright symbol and date.
    g. use HTML5 sematic elements to format the page.
3. **<u>In the contact.html page:</u>**
    a. Use lists and paragraph tags to display the contact information for the agency and the individual agents. Use list tags to show the at least 3 travels and their contact information. Feel free to add any other details to the page.
4. **<u>In the register page:</u>** set up a form that has fields for entering and submitting customer data.
5. Customize, experiment, and build extra features if you have time.

CSS

6. Download and link to an external stylesheet page that will reset your styles to defaults

7. Create an external stylesheet file and link it into your pages to provide custom style settings.
8. Define style settings that will set the font type, size, and color of the text in your tags. Experiment with different settings for different tag types.
9. Define styles to control alignment on some of your tags.
10. Set up borders on some of your tags.
11. Ensure that the input tags in your form all have a light background colour to make them more noticeable.
12. Use images, CSS elements and/or framework liberally to enhance your web pages.

JavaScript

In this exercise you will customize the web pages you have been working on earlier in this course and enhance the page to provide JavaScript functionality.

13. Set up a web form for customers to enter their name, address, city, province, postal code, and anything else that you want to include.
14. Modify your web form to ask for confirmation before submitting or resetting. Put a Click event handler on the submit and reset buttons of the form. When these buttons are clicked, confirm dialogs should pop open asking them if they really want to do this. If the user clicks "ok", the submit or reset continue, otherwise they should be cancelled.
15. Put a script tag in the head section of your main page and create two arrays. The first will have some image objects (limit this to a reasonable number, perhaps 4 or 5 images), and the second array will have strings with descriptions of each of the images. When the body of the page loads, loop through the arrays and get each image and the corresponding description and display them in a table row, one row per image.

Image	Description

16. Modify your code in part 3 such that instead of using two table cells per row, just create one cell with the description. Add a <div> tag, and add a mouseover event handler to each table cell. When the mouse pointer moves over the cell pass it's index number to a function. In the function, place an image tag into the <div> tag's innerHTML property. Set the <img> tag's source to the source of the image array's image object for that index.
*__Note:__
    - You can switch the logic such that an image appears first and the mouseover event triggers the display of the description.
    - You don't have to use a table for parts 3 and 4, you can use CSS layouts such as flexbox or grid.
17. Add an onfocus listener to each of the text fields in your form, that will display a paragraph with a description of the data the field requires (such as special data format instructions, etc.). Create a paragraph for each field and hide them when the page loads. Use CSS to set their size and position so they are in a stack.

Change CSS visibility settings to make the appropriate paragraph visible and the rest hidden whenever a field gets focus.

18. Add an array to your page that will have a URL of a website for each image in the images array. Try to find websites that fit well with the images that you are using.
19. Modify your image and description table (or any layout that you used). For each image, include a mouseclick handler that will call a function to open a new window and start a timer that will close the window after a few seconds.
20. Whenever you open a new window in step 2 (above), point the window (using hyperlink) to the website that corresponds to the image. If the hyperlink is not selected before the window times out, you can either continue to the link in the main window or do nothing.
21. Optional – If the basic stuff is boring to you, try this for extra challenge: Add a small image to your page that will be positioned using CSS. Use a timer to make the image move across the screen and back (get the window size from the window object and use it to determine when to reverse directions). If you want to add any extra features such as using a mouseover on the image to make it bounce away from the mouse pointer you are free to experiment. Have fun with it.

Feedback:

This lab is worth 30 marks. Your instructor will give you feedback.

Marking Rubric

Your mark is based on submitted work. Code will be examined using the following criteria.

PLEASE NOTE: Assignments are due on the date specified by the instructor. One mark will be subtracted if the files are submitted within one week of the due date. Files submitted after 1 week of the due date will have one mark subtracted per additional day beyond one week that the file is late.

Marks = 30 Possible Marks

Programming	**3**	**2**	**1**
Code Readability	Code is well-written, with consistent indentation, adequate white-space, and avoids long lines.	Code is readable but indentation, white-space, line length could all be improved.	Code is sloppy and hard to understand.
Syntax Errors	Programs compile cleanly.	Programs have some compile errors but an attempt has been made to fix them.	Programs are a long way from compiling.

Logic Errors	Programs are free of logic errors.	Programs have some logic bugs that could not be found, but were documented and an attempt was made to fix them.	Programs have major bugs.
Meeting Requirements	Does everything the assignment requested.	Does most of what the assignment requested.	Only partially completed.
Naming Standard	Follows object-oriented naming convention.	Partially follows naming convention.	Naming convention not followed.
Design	Programs are well-planned, well-organized, modular, easy to maintain or enhance.	Programs could be organized better, could be difficult to maintain or enhance.	Programs poorly organized, look like they were written without much planning.
Internal Documentation	Code is thoroughly documented	Documentation is partially done	Documentation is very sparse
Submission			
File Submission	Files are submitted to instructor by due date.	Files are submitted within 1 week of due date.	Files are not submitted within 1 week of due date. Beyond one week of lateness, one mark will be subtracted per additional day that the file is late.
Introduction of Submission	All files contain heading documentation identifying the author, date, course module and assignment.	Some of the required information is missing.	No identifying documentation.

Filename	Files are zipped into one file. Zip file name clearly indicates the course code, module/assignment name and student name.	File name is not entirely clear on the required information.	File name has none of the required information.