Web Application Development Project Submission 2022

Name: **Caoimhín Vallely**

Student ID: **G00398568**

Date: **17/05/22**

Home page/index page/start page: **login.html**

Project Requirements Implementation

ITEM 1	**Reference**
Allow the customer to enter their login details:	See: login.html
Login details validated (via a login screen) before receiving a summary of the order:	Validation performed using HTML5 and JavaScript. If successful, the user receives an alert, and is redirected to the homepage
Username:	charlie@parker.com
Password:	password!
Brief description of how this was implemented:	The entered username and password are taken in via a DOM query in a JavaScript function which checks them against hardcoded variables saved in the file. If correct, the user receives an alert which lets them proceed to the homepage. If the user enters the wrong username or password, an invalid email address, a password that's too short, or nothing at all, an alert with an appropriate message is displayed. The fields are then cleared, and the user can try again.

ITEM 2	Reference
Perform form validation through JavaScript or HTML to ensure that text fields are not empty, and a valid email address is entered	See: login.html and signup.html
Brief description of how this was implemented:	Most of the form fields from these 2 pages use the ***required*** attribute, so they will get a warning and the form won't submit if they are left blank. Equally the form will not submit if the email address is in the wrong format (***type="email"***) or if the date is in the wrong format (***type="date").*** If all the elements are correct, the user receives an alert and is redirected to the homepage. On the login page, I have also used a regular expression to test the validity of the email address. It tests whether the email address is in the correct format, i.e., contains several characters followed by '**@**' followed by more characters followed by '**.**' followed by more characters.

ITEM 3	Reference
Access and change HTML on the web page through the DOM;	See: login.html, homepage.html, poll.html, awards.html, sales.html, and signup.html
Brief description of how this was implemented:	On ***login.html*** there is a DOM query in a function which extracts user input and returns appropriate alerts. On ***homepage.html*** there are DOM queries which access and manipulate various style elements. On ***poll.html*** and ***awards.html*** we have some of the same DOM queries to change backgrounds, font colours and buttons There are also DOM queries in the function *drawChart()* which takes in user values (chart height, animation speed, colour) to manipulate the bar chart ***sales.html*** has the same function to change elements which uses DOM queries. On this page some text is also altered on the same click. On ***signup.html***, the submit function uses a DOM query to extract the users entered first name and use it in the alert message

ITEM 4	Reference

Access and change styling through the DOM;	See: homepage.html, poll.html, awards.html and sales.html
Brief description of how this was implemented:	These pages let the user click on buttons which change various features including backgrounds, font colours, button colours, and actual text. These are facilitated by DOM queries. The styling of the bar charts on ***poll.html*** and ***awards.html*** (size, colour elements, and animation speed) can also be manipulated by user input which is accessed via a DOM query in the relevant function. Some general text styling and formatting is also carried out via DOM query on ***poll.html***, ***awards.html*** and ***sales.html***.

ITEM 5	Reference
Demonstrate the use of events;	All pages
Brief description of how this was implemented:	***login.html*** has submit and reset buttons which return various alerts and results. On each subsequent page you have a navigation bar where there are clickable links to the other pages in the application. Hovering over each link changes its colour. The background colour and font colour of the current page are always highlighted. Each page has a 'sign-up' button which links to the ***signup.html*** page when clicked. There are '*hover*' effects on all of the '*rects*' in each of the bar charts on ***poll.html*** and ***awards.html***. There are clickable buttons which change stylings on ***homepage.html***, ***poll.html***, ***awards.html*** and ***sales.html***. ***sales.html*** has a button which creates a table of results. When the table is printed, a clickable link to the source data is displayed below. ***signup.html*** has a button which submits the form and returns an alert. ***listen.html*** has a selection of videos which can be watched from the page

ITEM 6	Reference
Contain two D3 data visualisations (e.g., Bar Chart) of your choosing *a. One from a CSV file* *b. One from an array*	See: *poll.html* and *awards.html*

Brief description of how this was implemented:	A D3 bar chart is created on ***poll.html*** from the csv file ***poll_results.csv***. The number of votes dictates the height of each '*rect'* while the 'artist' info and string version of votes are used as labels. Another D3 chart is created on ***awards.html*** using 2 arrays – a list of numbers (representing no. of awards) and another with artists'names.

ITEM 7	Reference
Both visualisations should allow the user to specify display settings, including an option to change colour, display size and animations	See: poll.html and awards.html
Brief description of how this was implemented:	The user can change various colour elements (bars, borders, text), animation speed, and height in pixels on both charts. This was implemented with DOM queries extracting the values and entering them into the D3 functions to create the charts.

ITEM 8	Reference
Have a minimum of 3 linked pages;	There are 7 linked pages: login.html

	homepage.html poll.html awards.html sales.html listen.html signup.html
Brief description of how this was implemented:	On successful login from ***login.html,*** the user is directed to ***homepage.html***. This is achieved through the *window.location.href* method. From there, all the other pages are accessible through the navigation bar using the <*a href*> method., The ***signup.html*** page is also linked via a button from each of the other pages using the code *onclick="window.location.href='/signup.html'"*

Additional information:

The application has consistent styling throughout. Most of the style elements are contained in the stylesheet (style_sheet.css) which is linked to each page. Other style elements are accessed and manipulated via DOM queries and user input.

There are images contained in the **Images** folder which are used on various pages with various stylings and formatting for the respective page. There are YouTube videos included on ***listen.html***, and a clickable external link contained on ***sales.html***. This opens a new tab with the source data for the table.