|  |
| --- |
| College LaSalle |
| Project - Internet Programming Technical Manual |
|  |
| Presented to: Juan Dario Silva Rodriguez |

|  |
| --- |
|  |

**Start by adding a short description of your project, and the languages (technologies) used**

Languages:

1. JavaScript
2. CSS
3. HTML
4. Grid layout
5. Flex Layout

**Presenting how the files/folders are organised.**

**the structure of the project is as follows**

* Index.html: On this page we find the login. Where we can access the website if our credentials are correct. If we do not have access we can go to the profile to create a new username and password. we can also observe the corporate image of the application.
* Style.css
* Index.js
* Candidates
  + Index.html : On this page you will find the main functionality of the application, which is to search by words in the list of candidates. These candidates will be automatically performed in a box
  + Style.css
  + Index.js
* Img
* Menu
  + Menu.html: In this section you will find information about the company and the services it provides, it also has an interaction where with a click these services will appear on the screen.
  + Style.css
* Profile
  + Index.html: In this section, users can update their profile and password
  + Style.css

**Present the functions and/or methods that you create or you did use into the web project.**

|  |  |
| --- | --- |
| Function or Method | Description |
| validate() | This application obtains the value of the input of the username and password and verifies if this value is inside the array login\_username and login\_password. If the username and password do not exist or do not match then the page will reload. |
| includes() | looks for the values inside the array it receives as a parameter |
| document.getElementById(id).value; | This method brings to the javascript file the value of the field identified with the id that it has as a parameter |
| window.open() | open the url that was entered as input |
| \* {    box-sizing: border-box;    margin: 0;    padding: 0;  } | in this css configuration we are eliminating all the predefined designs of css also we are expanding the scope of the entire html to the entire page |
| html {    font-size: 62.5%;    font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen, Ubuntu, Cantarell, 'Open Sans', 'Helvetica Neue', sans-serif;  } | In this configuration, the rem value is being configured, where by default the font-zise will become 1 rem. I do this to be able to use relative measures instead of absolute measures and thus be able to generate responsive web. also establishes roboto as the general source of the website |
| :root {    --navbar-color : #281432;    --morit : #79317E;    --rojit: #DE4859;    --amar: #EEA13C;    --white: #F2F2F2;    --black: #000000;    --h3title: #F2C94C;    --sm: 1.4rem;    --md: 1.8rem;    --lg: 2rem; | in this css configuration the color palette of the whole system is set and named as variables. We also set the small medium and large font size as variable. |
| display: grid; | With display grid I can use the most modern forms of web layout. Where it is organized by quadrants that adapt responsively to the screen. |
| grid-template-columns: 1fr 1fr; | The grid-template-rows and grid-template-columns properties are the primary CSS properties for establishing a grid layout, once the element is a grid context |
| grid-auto-rows: 1fr; | The grid-auto-rows property sets a size for the rows in a grid container.This property affects only rows with the size not set. |
| grid-auto-flow | The grid-auto-flow CSS property controls how the auto-placement algorithm works, specifying exactly how auto-placed items get flowed into the grid. |
| grid-area | the name of a grid area specified with grid-area |
| display: flex | The flex property is a shorthand property for: The flex property sets the flexible length on flexible items. |
| flex-direction: | Defines how flexbox items are ordered within a flexbox container |
| candidates\_name.forEach(function(item) | this function receives the names of the political candidates as an array and appends each name to each option in the data list. The result is that each of the candidates can be searched in the input only with matching of the letters |
| function get\_data(){ | this function receives as input the selection in the datalist mentioned above and renders it inside the main box without. With this function it was possible not to have to create several divs and hide them with the hide function. Because it automatically searches within the arrays and shows the user's choice |
| filter: contrast, brightness; | Change all images to black and white (100% gray) |
| transition: transform 250ms ease-in-out; | transition properties allow elements to change values over a specified duration, animating the property changes, rather than having them occur immediately. So, if we have, say, a box with a red background that we want to change to a green background when it is hovered, we can reach right for the transition property to move between background colors |
| $( ".menu-body-cards" ).hide(); $( ".menu-body-cards" ).show(); | The show() and hide() Method in jQuery is used to display the hidden and selected elements. |
| check\_pass() | This function receives the name of the password and its confirmation and checks that they are the same to allow the login to continue |
| setCustomValidity | If an input field contains invalid data, display a message: |
| querySelector | The querySelector() method returns the first element that matches a specified CSS selector(s) in the document. |
| .test() | The test() method tests for a match in a string.If it finds a match, it returns true, otherwise it returns false. |
| checkText() | This function checks if the user format is equal to that of an email, and if it does not meet this condition it allows it to send the form, if it does not disable the submit button |

**Present the difficulties that you have, what was the hardest and the easiest part of your project.**

The biggest difficulties was being able to structure the content of the page in a responsive way. Being able to put the entire design on the screen exactly as I had in mind was very difficult for me. That is why I had to investigate and learn about the latest technologies to layout web pages such as dislplay grid and flexbox.

The dom manipulation methods also seemed very difficult to me.

the easiest part for me was the javascript code because the logic is easier than the visual part