Marcelo Gaspar
drmpgaspar@gmail.com

Web Quiz Documentation

6th November 2019

Overview

This documentation aims to identify in an organized way all tasks divided by priorities and estimated execution time, and clearly show all the requirements requested to create this web test. The purpose of this project is to create a simple interactive test to assess participants' web development skills (HTML, CSS, JavaScript).

List of tasks to be performed:

01. Priority of each task
02. Title and description of each of them
03. Difficulty level
04. Estimated time for each task.
05. Record of incidents that were detected during project execution.
06. Record of lessons learned.
07. Project calendar
08. The Chronogram of the project.
09. Quality control measurements.
10. Quality metrics
11. Requirements documentation.
12. Risk documentation in the event that these exist
13. Documentation about the git WORKFLOW you are going to use
14. Documentation about the tools used in the project

Priority (1-3) 1 = low, 2 = moderate, 3 = high
A: Organization ---------- 3
B: Layout ---------- 2
C: Repository in git ---------- 3
D: Structuring in HTML ---------- 2
E: Work with localStorage ---------- 3
F: CSS styling ---------- 2
G: JavaScript functions ---------- 3
H: Test ---------- 3

A: Organization = structuring all tasks by priority and estimated execution time.
B: Layout = design container and color choice.
C: Repository in git = put the project into a git repository to organize changes during construction.
D: HTML Structuring = create entire site structure without third-party libraries
E: localStorage = is a JavaScript object that we use to store data in the browser
F: CSS styling = the artist part to give the site beauty. Make responsive without frameworks.
G: JavaScript functions:
+ Functions for storing user and score in localStorage.
+ Create a file .js with all questions in JSON format.
+ Fetching the questions in the created array.
+ Function that checks user response
+ Show score when finish questions
+ Function to increase the progress bar
+ Put the data of localStorage in the leaderboard
+ Function to sort the ranking in order of score
+ Create HTML elements only with JavaScript
H: Test = test in all browsers but the code has to be tested all the time.

Difficulty level (1-3) 1 = low, 2 = moderate, 3 = hard
A: Organization and Documentation ---------- 2
B: Layout ----------1
C: Repository in git ----------1
D: Structuring in HTML ---------- 2
E: localStorage ---------- 3
F: CSS styling ---------- 2
G: JavaScript functions ---------- 3
H: Test ---------- 2

Time (hours)
A: Organization and Documentation = 4h
B: Layout = 3h
C: Repository in git = 1h
D: Structuring in HTML = 4h
E: localStorage = 4h
F: CSS styling = 5h
G: JavaScript functions = 8h
H: Test = 3h
I: Incidents = 8h (20% total time)

Total time: 40h

Incidents

- Difficulty fulfilling the given time for each task
- Some tasks were underestimated
- Difficulty to make responsive without CSS framework
- Stop the project to study some localStorage concepts used
- Knowledge limitation in JavaScript
- Difficulty optimizing time for each task
- Limitation to implement certain features
- Difficulty working with localStorage and creating more elaborate javaScript functions

Lessons

- Better organize myself to accomplish the tasks
- I learned to prioritize the most important tasks
- I learned how to use localStorage
- Use git to our advantage
- Take questions with colleagues
- Make good use of JavaScript
- Better understand functions and create HTML elements with JavaScript
- Have patience and be humble

Project Calendar

Main tasks
06/11 -> Organization and Create repository in git
07/11 -> Create the 3 types of questions and study localStorage
08/11 -> Functions with JavaScript and create questions database
09/11 -> Make Styling with CSS to make responsive
10/11 -> Revise all code and make it cleaner + Test in browsers

TASK/ DAY	WED 06/11	THU 07/11	FRI 08/11	SAT 09/11	SUN 10/11
A	X	X			
B	X				
C	X				
D	X	X	X		X
E		X	X	X	X
F			X	X	X
G					X
H					X

Chronogram of the project

06/11 - Organization and division of tasks by priorities
- Start documentation
- Put repository in git
- Start HTML basic structuring
- Study localStorage

07/11 - Continue quiz structure
- Work with git
- Create the type of questions (input)
- Implementation of user data in localStorage
- Start basic styling with CSS

08/11 - Functions with JavaScript and create questions database
- Continue styling with CSS
- Continue work with git
- Continue work whit localStorage

09/11 - Continue create JavaScript functions
- Complete final details of the structuring
- Complete final details of the styling
- Finalize documentation

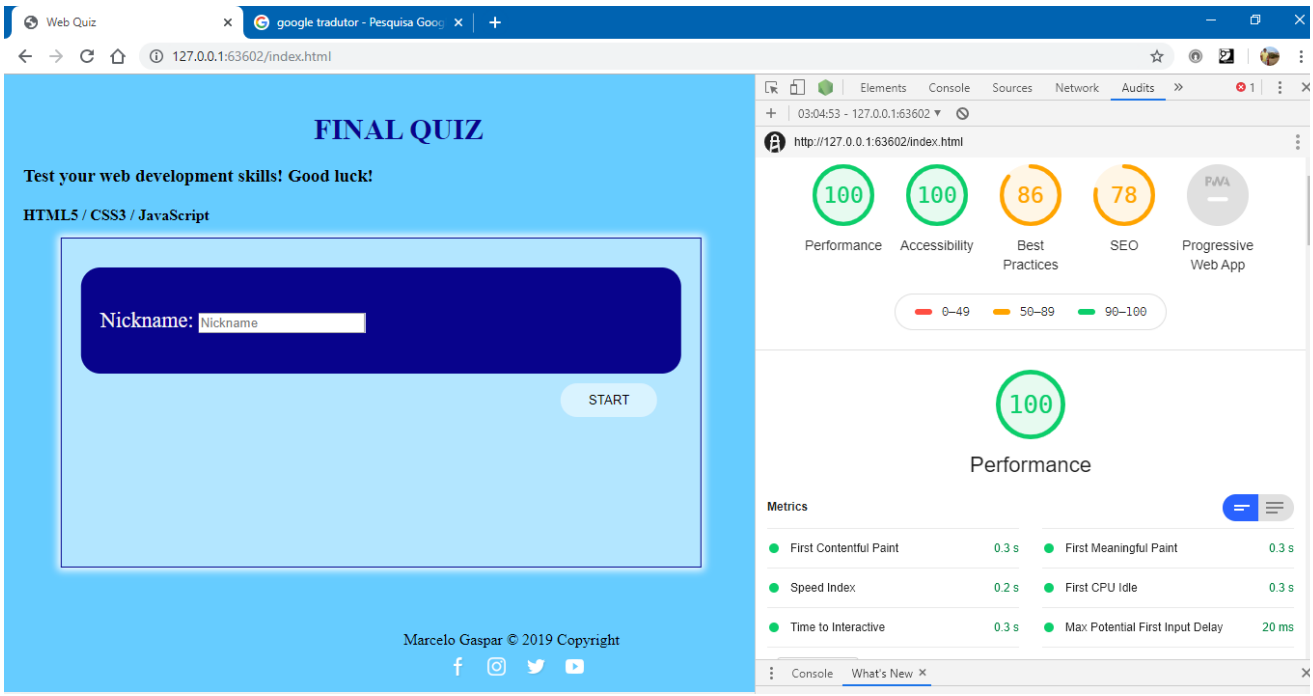
10/11 - Test in browsers
- Push to GitHub repository

Quality control measurements
The quality control measurements are used to analyze as well as evaluate the quality of the different processes involved in a project against the standards of the organization or on the requirements specified during the project management planning.

- Is the HTML code properly validated by the W3C ?
- Are the HTML, JS and CSS files properly formatted ?
- Did you use clean code?
- Is the site responsive and easy for the customer to understand?
- Does the quiz have all the specifications requested?
- Does the quiz look good when displayed in any of the common browser clients?
- Do the commit messages properly explain the development flow of the project?
- Have all functions been tested throughout the project realization process?

Quality metrics
Google Chrome Developer Tools (Audits)
Identify and fix common problems that affect your site's performance, accessibility and user ex perience
- Performance
- Progressive Web App
- Best practices
- Accessibility
- SEO

Metrics:
First Contentful Paint----------0.3 s
Speed Index----------0.2 s
Time to Interactive----------0.3 s
First Meaningful Paint----------0.3 s
First CPU Idle----------0.3 s
Max Potential First Input Delay----------20 ms



Project requeriments

The questionnaire will have to contain at least 20 questions.

- The format of the answers can be divided into:
  - Select single answer from multiple options (radio)
  - Select multiple answers (checkbox)
  - Select from a drop-down menu
- The first screen that will be displayed will ask for a nickname, which will be saved in the **localStorage** and will be shown in the ranking once the questionnaire is finished.
- There will be a progress bar that will measure the percentage of questions answered regarding the total (this bar should be displayed on all pages and should be updated after answering each question).
- The progress bar will be divided into sections (eg, HTML, CSS, JS ...) and should be updated dynamically every time a question with an CSS3 animation is answered.
- After answering each question the result will be stored in **localStorage**.
- There must be a call to action to start the questionnaire. This start creates a new record in **localStorage** to be able to store multiple questionnaires that can be ordered later by the final scoring. It is important that you associate a scoring to each question in order to obtain the final result.
- The questions will be distributed on at least 3 separate pages. Each time you go to the next page, you must save all the information from the previous one in **localStorage**(the answers and the current status of the progress in the questionnaire, which would be the percentage of answers made with respect to the total answers).

❍All the evolutionary will be carried out locally and without a database, using HTML, CSS and Javascript (NO third-party libraries can be used)

❍You must use GIT

❍Write clear and concise commits, should not extend more than 50 characters.

❍It has to be **responsive** and you must get it without using third-party libraries.

❍When moving from one question to another, an animation made with CSS3 will be displayed.

❍At the end of the questionnaire, you must show a final screen with your result and the final ranking indicating your score and that of the other records that have been previously made. The ranking table is filled with the records stored in the local storage. This table can be ordered from highest to lowest score by a button.

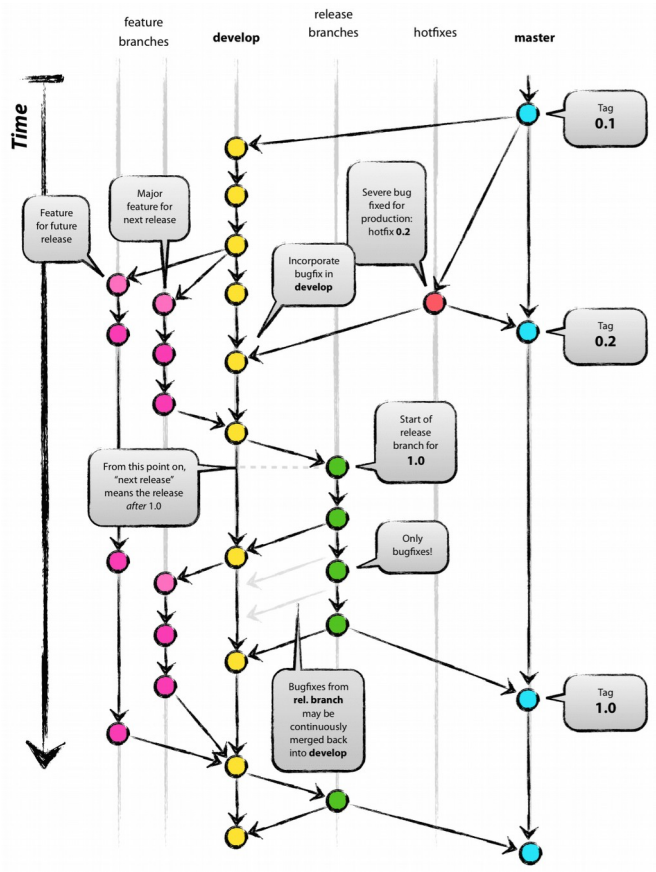
Risk documentation

Incidents (disease, technical failure, poor performance, unrealistic deadlines) may change:

+ Statement of work (SOW)
+ Work breakdown structure (WBS)
+ Budget
+ Schedule
+ Execution plan

Git Workflow

Described by Vincent Driessen in 2010.



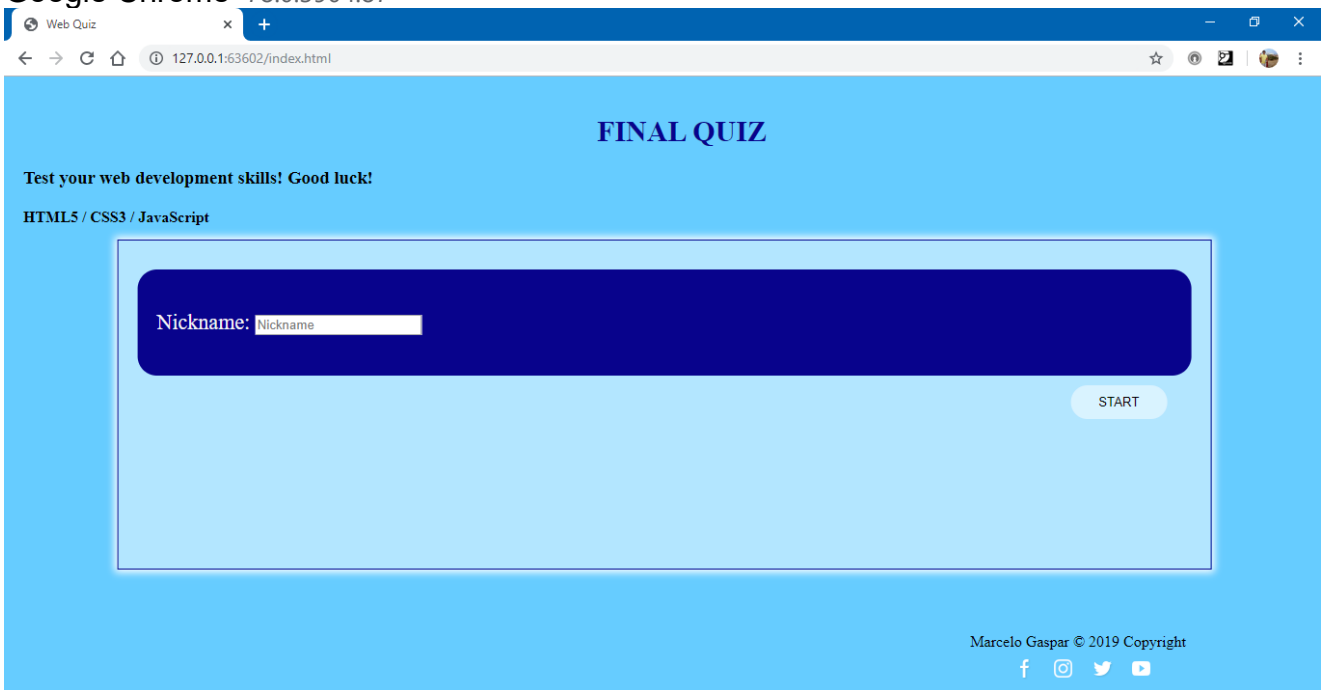
I used 3 branches:
- Master
- Develop
- Feature

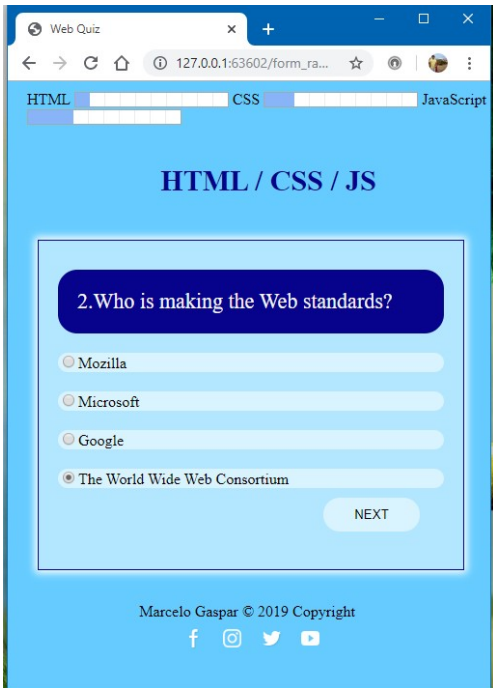
+ “Master” is always ready to be released on LIVE, with everything fully tested and approved (production-ready).

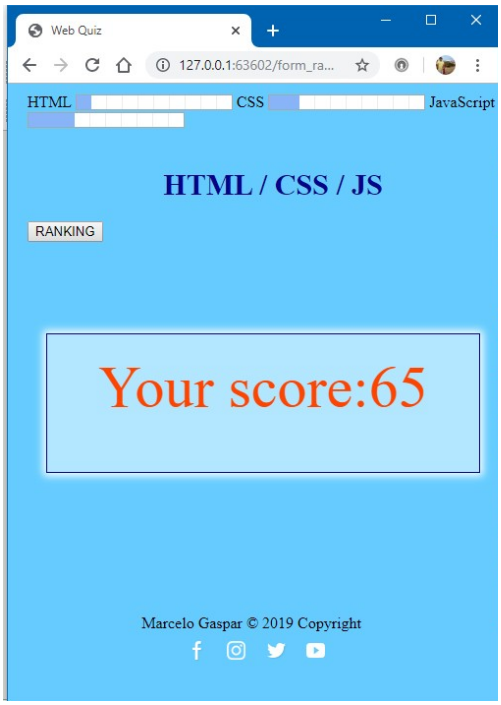
+ “Develop” is the branch to which all feature branches are merged and where all tests are p erformed. Only when everything’s been thoroughly checked and fixed it can be merged to th e Master.

Project Tools
- Visual Studio Code
- Git and GitHub
- Google Chrome Developer Tools

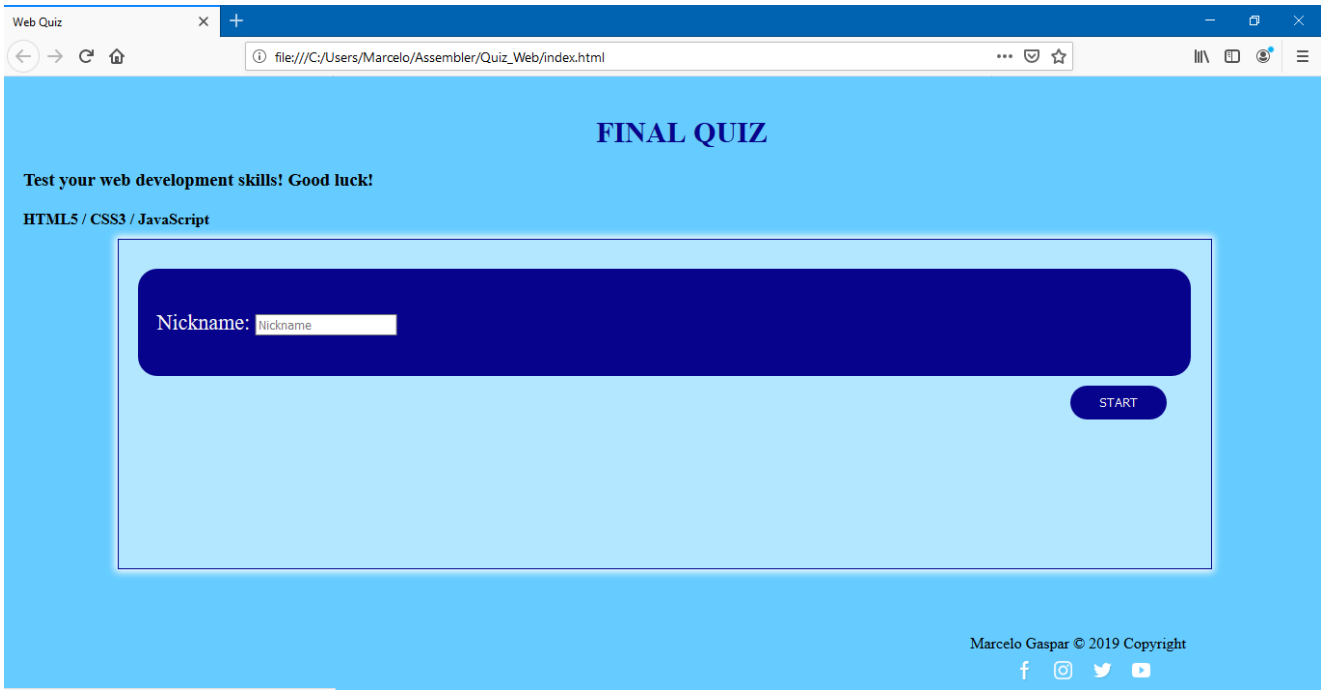
Google Chrome 78.0.3904.87

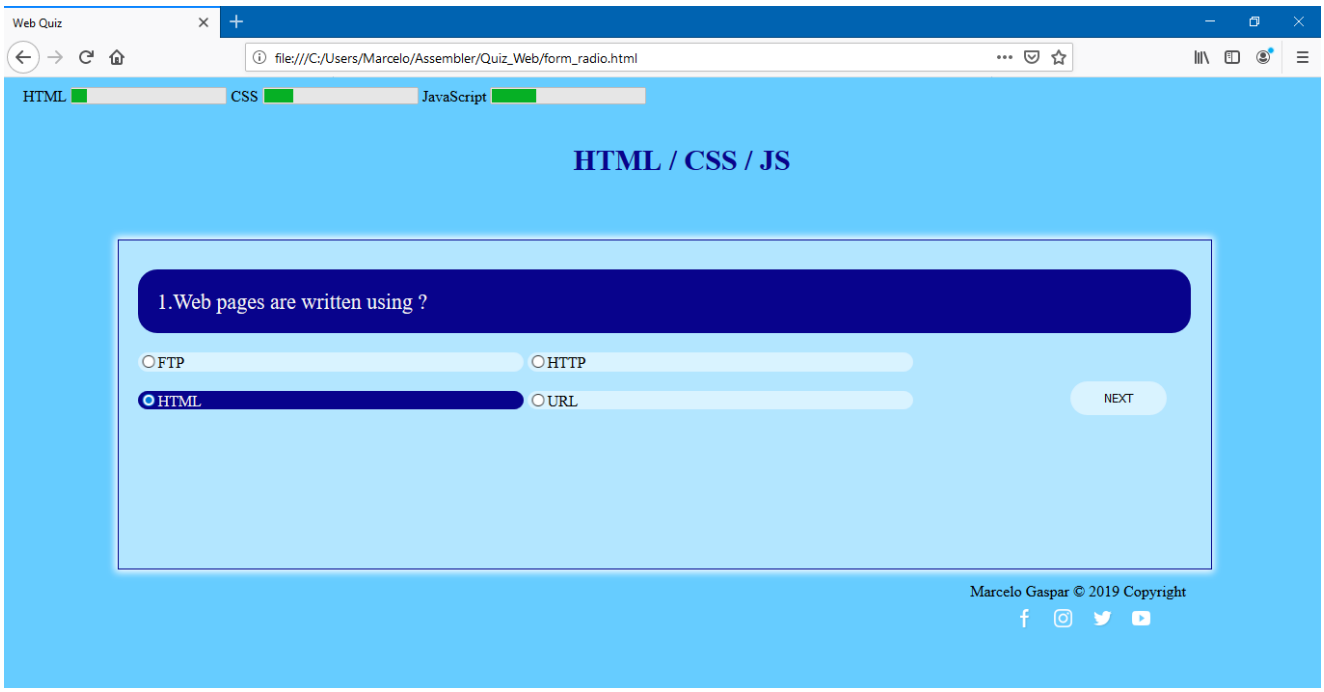




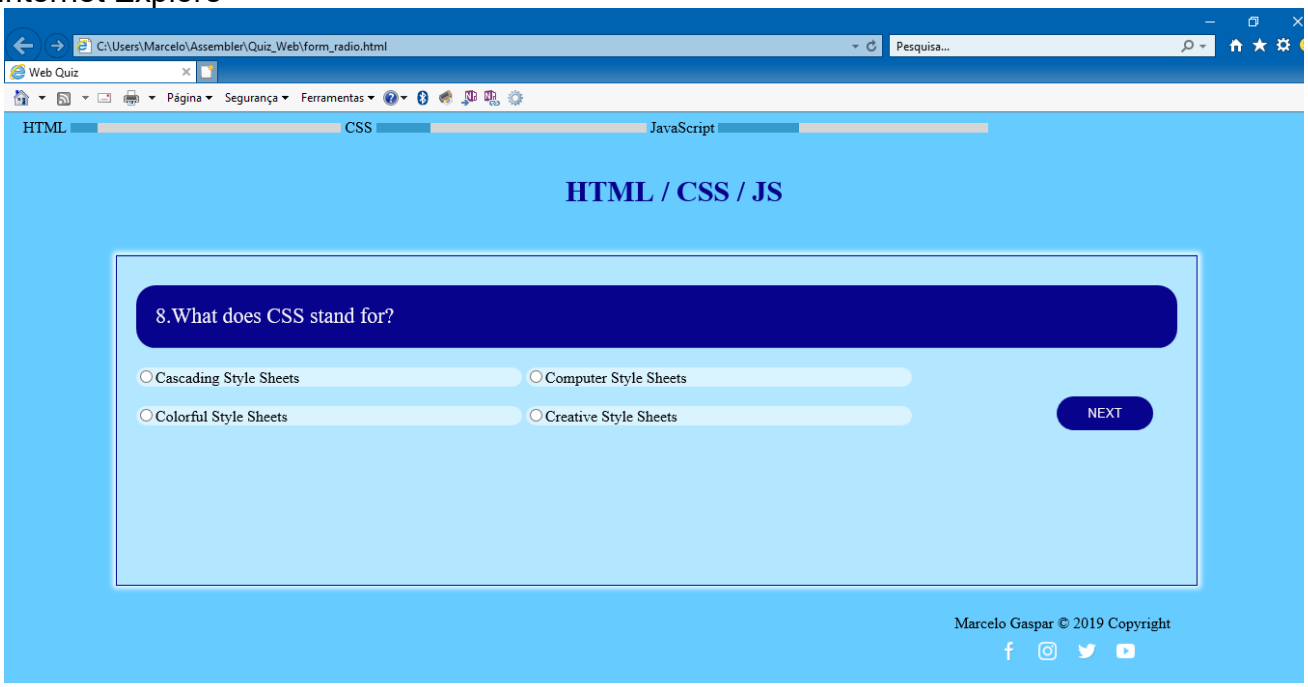


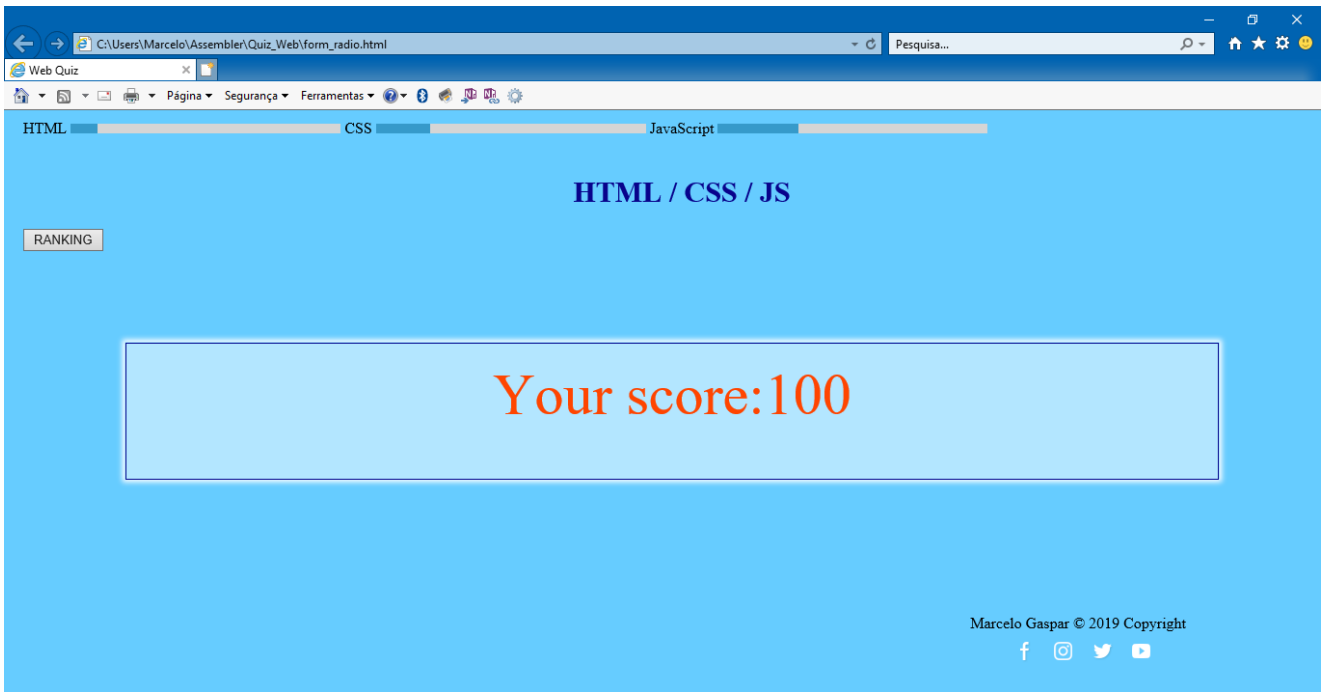
Mozilla Firefox





Internet Explore





Safari 13.0.3

References to make the project:
https://developer.mozilla.org/es/docs/Web/API/Window/localStorage
https://www.w3schools.com/
https://stackoverflow.com/