WEB programming
Lab 02: CSS

Instructions:

- The aim of these labs is to teach you how to find information on your own. The teacher's role is to support you in this process and guide you to the most relevant resources.
- This tutorial is to be completed alone.
- Before starting each lab activity, you'll need to do some preliminary reading to get started and/or deepen the knowledge you've acquired during the lecture.
- To help you with your practical work, you'll need to conduct research on Mozilla Developer Network (MDN), Stack Overflow, W3Schools or any other relevant ressources.
- ChatGPT can be used to search for errors in your code.
- Give priority to searches in English.
- You can choose between using VSCode, a classic editor (SublimeText or Notepad++) or Codepen.
- The complete Code must be submitted on Moodle before Lab3

Learning outcomes :

- At the end of this lab activity, you should be able to :
  - o DOM: adding style to a web page
  - o CSS: tag selectors, properties and values
  - o Using ID, class and attribute selectors

Green IT 

The aim is to implement an informative website presenting Green IT and some practical tips for web development, as well as a Green IT knowledge test.
This site has 3 web pages:

- Home.html: this page provides information on green IT.
- GreenITSiteWeb.html: this page provides information on the carbon footprint of websites and some practical tips for web development.
- QuizGreenIT.html: this page tests a visitor's knowledge of Green IT.

Starting point: Source code from Lab1

To get to grips with CSS concepts, please use the source code from Lab1. Alternatively, you can use the Source code given as a correction, which is available on Moodle.
Note: We strongly recommend that you create a **copy of Lab1 as Lab2** to keep an up-to-date copy of your work.

Step 1: Adding a style file

Pre-reading:

- MDN :Using CSS in a WEB page

1- Open the Lab2 folder and create a subfolder named **css** in the same directory as the **html** subfolder.
2- In css, create a **styles.css** file
3- In the <head> tag of html files, add the link to the style file you created using the tag below.

```
<link rel= "stylesheet" href="../css/styles.css">
```

Step 2: Basic css properties, inheritance and pseudo-classes.

Further reading :

- MDN: CSS basics
- MDN: Top-down combiners
- DND: Neighbouring combiners
- MDN: Pseudo-classes

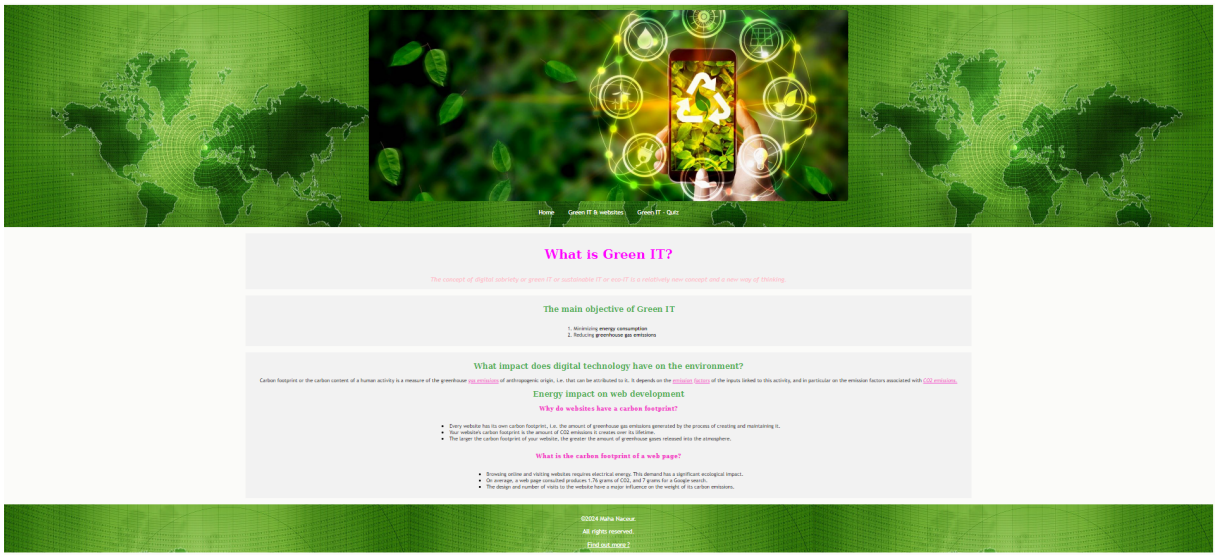
Page : home.html

1. Formatting common to all pages (**body** selector) :
    a. Change the background color of the 3 pages using the ***background-color*** property.
    b. Change the text font using the ***font-family*** property
    c. Justify the content using the ***justify*** value.
2. **header** and **footer** :
    a. Add a background image for the <header> and <footer> (image of your choice) by setting the url (link to your image) as the value of the ***background-image*** property.
    b. Add 15px ***padding*** to page content
    c. Center the alignment of <header> and <footer> elements
3. **Section** tags :
    a. Add a fixed ***margin-bottom*** and ***margin-top***.
    b. Add a background color
    c. Add fixed ***padding*** to all sections.
4. **h1, h2 and h3** :
    a. Give a color for <h1> tags

b. Change the color of <h2> in all sections.
c. Give a different color to <h3> tag that are positioned at the same level as h2 section headings. To do this, use the ~ symbol.

5. Navigation bar :
   a. Block <nav> elements using ***display***
   b. Add a ***padding*** and a ***margin*** of 10px
6. Links:
   a. For all links :
      i. Remove the blue underline that appears using ***text-decoration-line***
      ii. Add a background color on mouseover (pseudo-class***: hover***).
      iii. Set text to white using ***color***
      iv. Set borders with ***border-radius*** property
   b. For links in the navigation bar :
      i. Increase size with ***font-size***
      ii. Add a different color to the text when hovering over each link (***:first-child*** and ***:nth-child()*** pseudo-classes)
      iii. Add a different background color to each link hover.
   c. For links contained in a section :
      i. Set text to pink using ***color***
      ii. Change the font to italic using ***font-style***.
7. Header image:
   a. Add width and height to the image in the header that adapts to the screen size using ***height*** and ***width.***
   b. Round the corners of the image using the ***border-radius*** property.

In addition to what you've just set up, I invite you to explore new css properties.



Step 3: Stylize content using ID, class and pseudo-class selectors

Further reading

- MDN : Class selectors
- ID selectors
- Attribute selectors

Note: the styles.css file is common to all pages. In this section, we'll be working with new selectors.

1- In the home.html page, add the *id = "definition"* attribute to the p tag containing the following paragraph:

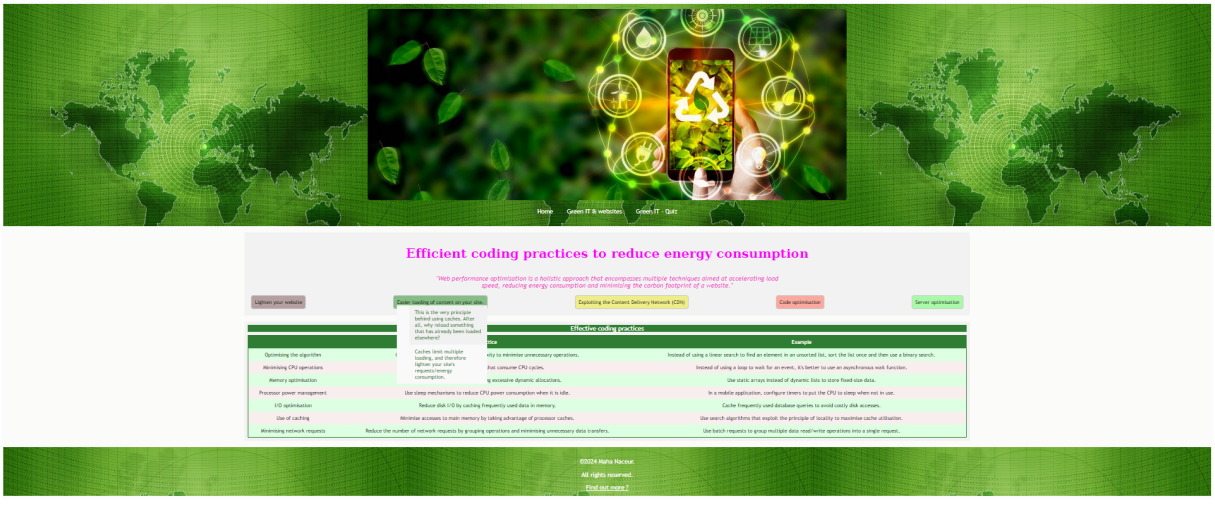
```
<p id ="definition">
    The concept of digital sobriety or green IT or sustainable IT or eco-IT
    is a relatively new concept and a new way of thinking.
</p>
```

2- In the GreenITSiteWeb.html page, add the *id="citation"* attribute to the p tag containing the following paragraph:

```
<p id="citation">
    "Web performance optimisation is a holistic approach that encompasses multiple techniques
    aimed at accelerating load speed, reducing energy consumption and minimising the carbon
    footprint of a website."
</p>
```

3- Using the ID selectors (#citation and #definition), add properties for formatting the two paragraphs. Example: change text color, italicize text, etc.
N.B.: remember to factor when properties are common to both paragraphs.

4- The idea here is to transform the list on the GreenITSiteWeb.html page into colored post-it notes, hiding the sub-lists and making them visible only on mouse-over.



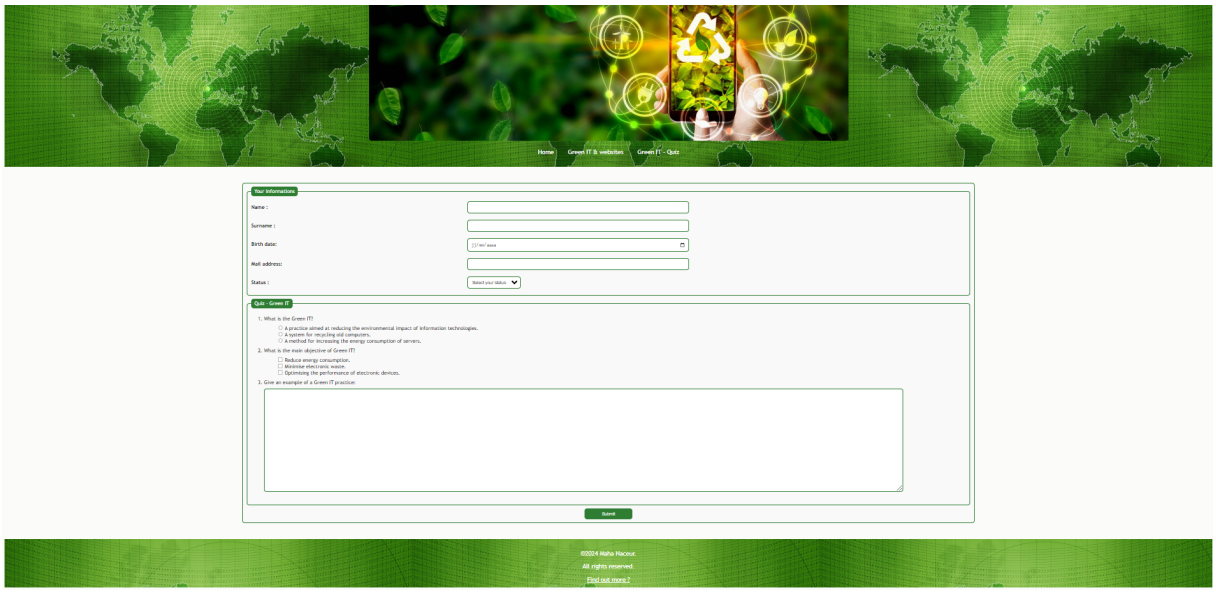
a. Give the *id = "postit"* attribute to the ol tag and add the following properties:
    i. Using the *display* property to specify the desired display type for the postit element, set the value to "*flex*" to create a flexible container.
    ii. Use the *flex-wrap = "wrap"* property to move to the next line when horizontal space is insufficient.
    iii. Remove bullets from the postit list using the *list-style-type* property.
    iv. Using the justify-content property, add a space between elements.
    v. Set *padding* and *margin* to 0.
b. Give the *class = "postit-item"* attribute to items in the postit list, and add the following properties:
    i. A *background color*.
    ii. A border
    iii. Apply *position :relative* to define the position of elements with top, bottom, left and right properties.
    iv. *Padding* and *margin* to space out border content
c. Add the following properties to the ul sub-lists contained in the various items:
    i. *display : none* to hide elements
    ii. *position : absolute*
    iii. *min-width* to define the minimum width of a sub-list.
    iv. *Z-index to* control the position of sub-lists in terms of depth on the Z axis perpendicular to the screen.
d. When hovering over postit-item elements, display ul sub-lists using *display : block*
e. Add a style for sub-list items: color, padding, etc.
f. Using the *nth-child()* pseudo-class, give each item in the postit list a different color.

Stage 4: Autonomy

Using the concepts you've learned in steps 2 and 3, add style to the table on the GreenITSiteWeb.html page and to the form on the QuizGreenIT.html page. Don't forget to style the elements on the about.html page.

Note: you can also use other selectors such as the attribute selector, e.g. *input[type = "text"]*.

Here's an example of what I got for the form.



Step 4: Make our site responsive

Further reading

- MDN : Viewport
- Media queries
- CSS Device Adaptation

1- At the head of your html document, add the "viewport" tag below:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

2- Check how rendering reacts to different page widths with the "Adaptive View" tool (Firefox)/"".Toogle Device Toolbar"(Chrome) in your browser's development console.