# LAB1: HTML, CSS, Static web development with HTML5 and CSS3

## I. HTML

### 1. Minimal structure


***Explanation:***

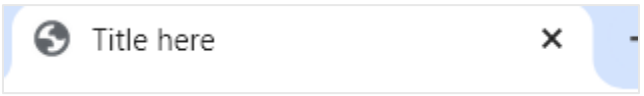
**<!DOCTYPE html>:** This declaration defines the document type and version as HTML5. It ensures that the browser interprets the document correctly.

**<html>:** This tag wraps the entire HTML document, indicating the start of the HTML content.

**<head>:** This section contains metadata about the HTML document, such as its title, character set, and links to external resources like stylesheets or scripts. It doesn't directly affect the visible content of the webpage.

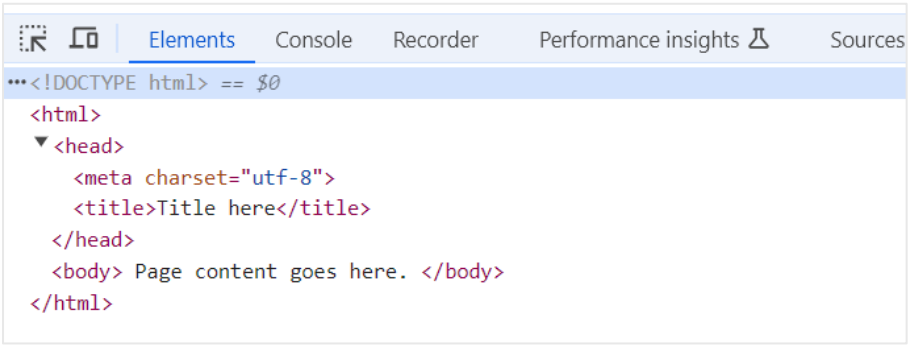
**<meta charset="utf-8">:** This tag defines the character encoding of the document as UTF-8, which supports most characters from any language, ensuring proper text display.

**<title>:** The text between the <title> tags is the title of the webpage, displayed on the browser tab. It helps identify the page.




**<body>:** This section contains the content that is displayed in the browser. In this case, the content is "Page content goes here."

When using the **F12** key, you can see how the browser 'reads' and 'understands' the code:




***Questions:***

1. Change the text inside the **<title>** tag to **"My First Web Page"**. What difference do you notice in the browser tab?

2. Add a **<link>** tag inside the **<head>** section with this code:

```
<link rel="icon" href="https://static.xx.fbcdn.net/rsrc.php/yT/r/aGT3gskzWBf.ico">
```

   How does this change the icon of the page?

3. Replace the content between the **<body>** tags with a heading, like **<h1>Welcome to My Site</h1>**. How does this affect the visible content in the browser?

4. Add a **<p>** tag inside the **<body>** section with some paragraph text. For example:

```
<p>This is my first paragraph.</p>
```

   How does this change the display of the page?

## 2. Basic Element HTML

An HTML element is the basic building block of a webpage. Every element consists of:

- Opening Tag: This marks the start of the element. It includes the element name inside angle brackets.

- Content: This is the text or nested elements inside the element.

- Closing Tag: This marks the end of the element, using the same name as the opening tag but with a forward slash (/).

***Example:***

```
<p>This is my first paragraph.</p>
```

** :** Opening tag (defines a paragraph).

**This is a paragraph.**: Content.

** :** Closing tag (ends the paragraph).

### a. Heading Element (<h1> - <h6>)

Used to define headings on a page. There are six levels of headings, from <h1> (the largest) to <h6> (the smallest).

```
<h1>Heading Level 1</h1>
<h2>Heading Level 2</h2>
```

### b. Paragraph Element ( )

Defines a paragraph of text. It automatically adds space before and after the text.

```
<p>This is a paragraph of text.</p>
```

### c. Link Element (<a>)

Creates hyperlinks. The href attribute specifies the destination URL.

```
<a href="https://www.google.com">Google</a>
```

### d. Image Element (<img>)

Used to display images. The src attribute specifies the image source (URL or file path), and alt provides alternative text.

```
<img src="image.jpg" alt="Description of image">
```

## 3. Attributes in HTML Elements

Attributes provide additional information about elements. They are always added inside the opening tag.

**Syntax:** <element attribute="value">Content</element>

**Common Attributes:**

**href** for links **(<a>)**

**src** for images **(<img>)**

**alt** for images (alternative text)

**class** and **id** for identifying or styling elements

Example with attributes:

```
<a href="https://www.example.com" target="_blank">Example</a>
```

In this case:

**href**="https://www.example.com" defines the link destination.

**target**="_blank" opens the link in a new tab.

**4. Self-Closing Elements**

Some elements don't have closing tags because they are self-contained.

Examples include:

**<img>** for images.

**<br>** for line breaks.

**<hr>** for horizontal lines.

**5. Nesting HTML Elements**

HTML elements can be nested inside each other, meaning you can put elements within other elements.

Example of nested elements:

** **This is a paragraph with a ****bold**** word.** **


In this case:

The ** ** element contains the text.

Inside the ** ** element, there's a **** element to bold part of the text.

**6. Table Element**

HTML tables are used to display data in rows and columns. The table is created using several nested elements, each with a specific role.

a. Basic Table Structure

A simple table contains:

**<table>**: The container for the table.

**<tr>**: Defines a row in the table (table row).

**<th>**: Defines a header cell in a table (table header).

**<td>**: Defines a data cell in a table (table data).

Example of a Simple Table:

```
<table>
  <tr>
    <th>First Name</th>
    <th>Last Name</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>John</td>
    <td>Doe</td>
    <td>30</td>
  </tr>
  <tr>
    <td>Jane</td>
    <td>Smith</td>
    <td>25</td>
  </tr>
</table>
```

Result:

| First Name | Last Name | Age |
|---|---|---|
| John | Doe | 30 |
| Jane | Smith | 25 |

**Explanation:**

**<table>**: Starts the table.

**<tr>**: Creates a new row.

**<th>**: Defines a header cell, usually bold and centered by default.

**<td>**: Defines a regular cell with table data.

b. Table Attributes

You can style tables using various attributes:

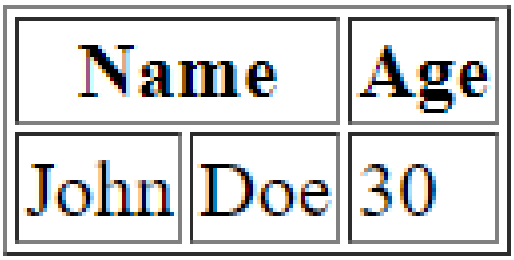
**border**="1": Adds a border to the table.

**colspan**: Allows a cell to span multiple columns.

**rowspan**: Allows a cell to span multiple rows.

Example with colspan:

```
<table border="1">
  <tr>
    <th colspan="2">Name</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>John</td>
    <td>Doe</td>
    <td>30</td>
  </tr>
</table>
```




**7. Form**

HTML forms are used to collect input from users, such as text, passwords, selections, and buttons. They are key components for gathering data on websites.

a. Basic Form Structure A form consists of:

**<form>**: The container for all form elements.

**<input**>: The form input field.

**<label>**: Describes what an input field is for.

**<button>** or **<input type="submit">:** A button to submit the form.

Example of a Simple Form:

```html
<form action="/submit" method="POST">
  <label for="fname">First Name:</label>
  <input type="text" id="fname" name="fname"><br><br>

  <label for="lname">Last Name:</label>
  <input type="text" id="lname" name="lname"><br><br>

  <input type="submit" value="Submit">
</form>
```

First Name:

Last Name:

Submit

**10. Inline and block level**

HTML elements are categorized into inline and block-level elements, each having different behavior in terms of layout and structure.

a. Block-Level Elements

Block-level elements start on a new line and take up the full width available (by default). They create a "block" in the document and can contain other block-level or inline elements.

Common Block-Level Elements:

** **: A generic container for grouping elements.

**<h1> - <h6>**: Headings.

**<p>**: Paragraphs.

**<ul>** and **<ol>**: Unordered and ordered lists.

**<table>**: Tables.

Example of Block Elements:

```html
<div>
  <h1>Main Heading</h1>
  <p>This is a paragraph inside a block element.</p>
</div>
```

b. Inline Elements

Inline elements do not start on a new line and only take up as much width as necessary. They are usually found inside block-level elements and are mainly used to style parts of text or content.

Common Inline Elements:

****: A generic inline container for text.

**<a>**: Anchor tag for links.

**** and ****: Bold and italic text.

**<img>**: Inline images.

Example of Inline Elements:

```
<p>This is a <strong>bold</strong> word inside a paragraph.</p>
<a href="#">This is a link</a>
```

## II. CSS

CSS is used to style HTML elements by adding design, formatting, and layout rules. Let's break down some key concepts.

### 1. CSS Declarations

A CSS rule consists of:

Selector: Specifies the HTML element to be styled.

Declaration Block: Contains one or more declarations.

Declaration: Consists of a property and a value separated by a colon (:).

Example of a CSS Rule:

```
p {
    color: blue;
    font-size: 16px;
}
```

Selector: p (applies styles to all **<p>** elements).

Declaration Block: Contains multiple declarations (**color** and **font-size**).

Declaration: **color**: blue; and **font-size**: 16px;

### 2. Adding CSS to HTML

There are three ways to apply CSS to an HTML document:

a. Inline CSS Directly within the element using the style attribute.

```
<p style="color: blue;">This text is blue.</p>
```

b. Internal CSS

Placed within the **<style>** tag inside the **<head>** section.

```
<head>
  <style>
    p {
      color: blue;
    }
  </style>
</head>
```

c. External CSS

Written in a separate **.css** file and linked to the HTML document.

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

**3. CSS Selectors**

CSS selectors are used to target specific elements for styling.

a. Element Selector

Selects all elements of a given type.

```
p {
    color: red;
}
```

b. Class Selector

Targets elements with a specific class attribute. Prefixed by a dot (.).

```
.red-text {
    color: red;
}
```

c. ID Selector

Targets an element with a specific id attribute. Prefixed by a hash (#).

```css
#main-heading {
    color: green;
}
```

```html
<h1 id="main-heading">Main Heading</h1>
```

**4. Formatting text**

Formatting Text CSS provides various properties to format text and control typography.
a. Text Properties

**font-family**: Specifies the font family.

**font-size**: Sets the size of the text.

**font-weight**: Controls the boldness of the text.

**text-align**: Aligns the text (left, right, center, justify).

**text-decoration**: Adds decorations like underline, overline, or line-through.

Example:

```css
h1 {
    font-family: 'Arial', sans-serif;
    font-size: 24px;
    font-weight: bold;
    text-align: center;
}
```

b. Line Height and Spacing

**line-height**: Sets the space between lines of text.

**letter-spacing**: Adjusts space between letters.

**word-spacing**: Adjusts space between words.

**5. Color and background**

CSS: Padding, Borders, and Margins

These properties control the spacing around elements.

a. Padding The space between the content and the element's border.

```
p {
    padding: 20px;
}
```

b. Border Defines the border around an element.

```
div {
    border: 1px solid black;
}
```

c. Margin The space outside the element, separating it from other elements.

```
h1 {
    margin: 30px 0;
}
```

**6. Floating, position, stacking order**

a. Floating Elements Used to **float** elements to the left or right.

```
img {
    float: left; margin-right: 10px;
}
```

b. Positioning Elements

**static** (default): Normal

**relative**: Positioned relative to its normal position.

**absolute**: Positioned relative to its nearest positioned ancestor.

**fixed**: Positioned relative to the viewport.

```
div {
    position: absolute; top: 50px; left: 100px;
}
```

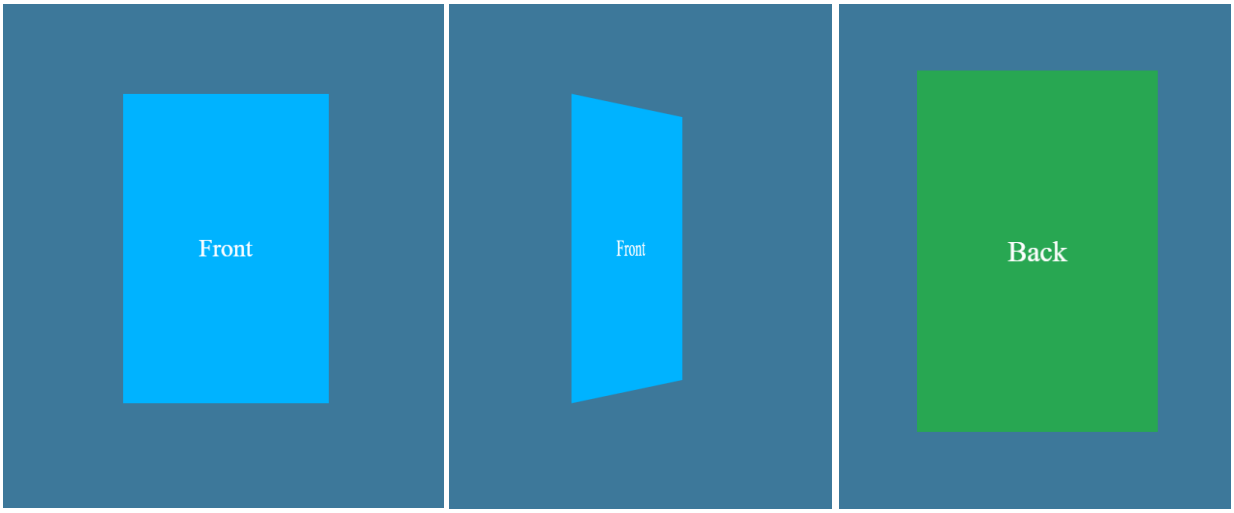
c. Z-Index (Stacking Order) Controls the stacking order of overlapping elements. Higher **z-index** values appear on top

```css
div {
    position: absolute;
    z-index: 10;
}
```

**III. Exercises:**

**1. 3D card flip effect**

Create a card with two sides, front and back. When hovering the mouse pointer over the card, it will flip over.




*Hint:* you can rely on the following HTML snippet to implement.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <div class="card">
        <div class="inner-card">
            <div class="card-face front">
                Front
            </div>
            <div class="card-face back">
                Back
            </div>
        </div>
    </div>
</body>
</html>
```

Background color: #3d789a
Background color of front face: # 00b3ff
Background color of back face: # 28a752

**2. Falling Triangles Animation**

Create green triangles that fall freely and disappear when they reach the bottom of the page.

*Hint:* This exercise involves using a bit of JavaScript. (color: # 228B22)

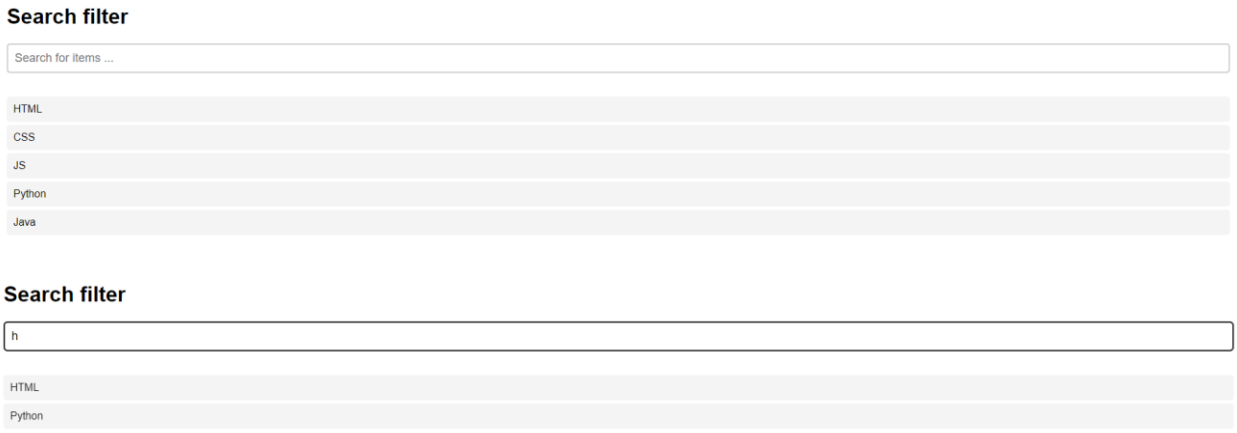
### 3. Rotating text

*Hint:* you can rely on the following HTML snippet to implement.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <link rel="stylesheet" href="style.css">
    <title>Document</title>
</head>
<body>
    <div class="rotate-point">
        <div class="carousel">
            <div class="cell" style="--i: 0;">r</div>
            <div class="cell" style="--i: 1;">o</div>
            <div class="cell" style="--i: 2;">t</div>
            <div class="cell" style="--i: 3;">a</div>
            <div class="cell" style="--i: 4;">t</div>
            <div class="cell" style="--i: 5;">i</div>
            <div class="cell" style="--i: 6;">n</div>
            <div class="cell" style="--i: 7;">g</div>

            <div class="cell" style="--i: 8;"> </div>

            <div class="cell" style="--i: 9;">t</div>
            <div class="cell" style="--i: 10;">e</div>
            <div class="cell" style="--i: 11;">x</div>
            <div class="cell" style="--i: 12;">t</div>



            </div>
        </div>
    </div>
</body>
</html>
```

## 4. Search filter

You will create a search bar, and as search characters are entered, irrelevant results will be filtered out.




*Hint:* you can rely on the following HTML snippet to implement and this exercise involves using a bit of JavaScript.

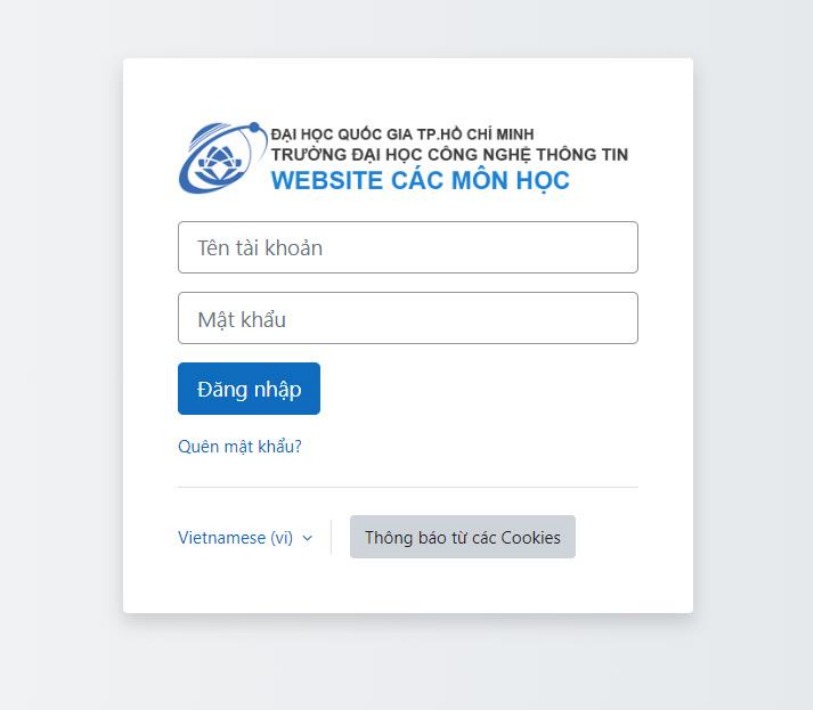
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <link rel="stylesheet" href="style.css">
    <title>Document</title>
</head>
<body>
    <h1>Search filter</h1>
    <input type="text" id="searchInput" placeholder="Search for items ...">
    <ul id="itemList">
        <li>HTML</li>
        <li>CSS</li>
        <li>JS</li>
        <li>Python</li>
        <li>Java</li>
    </ul>
    <script src="ex4.js"></script>
</body>
</html>
```

## IV. Homeworks:

In the homework, you will clone the pages below. Your grade will be proportional to the similarity between your work and the provided images.

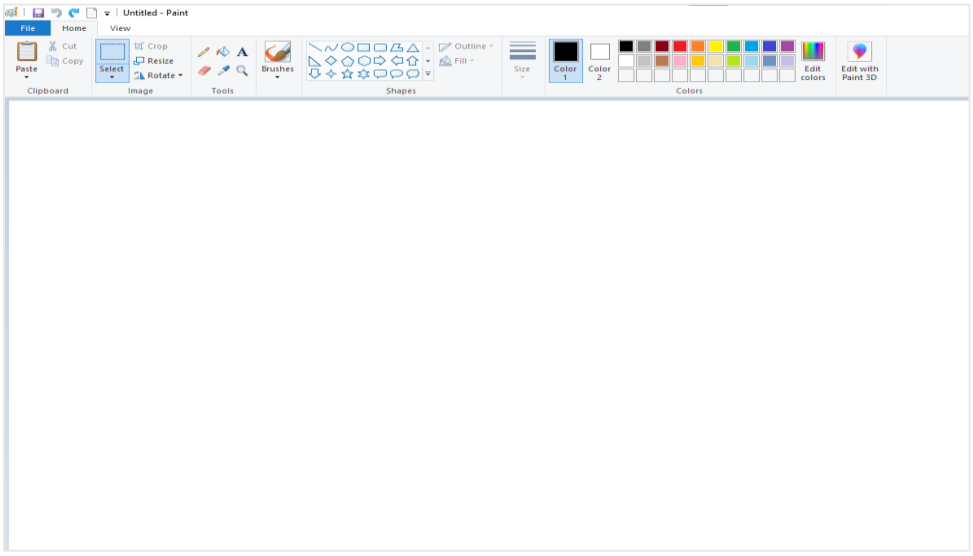
### 1. UIT courses login page

Link to website: *https://courses.uit.edu.vn/login/*

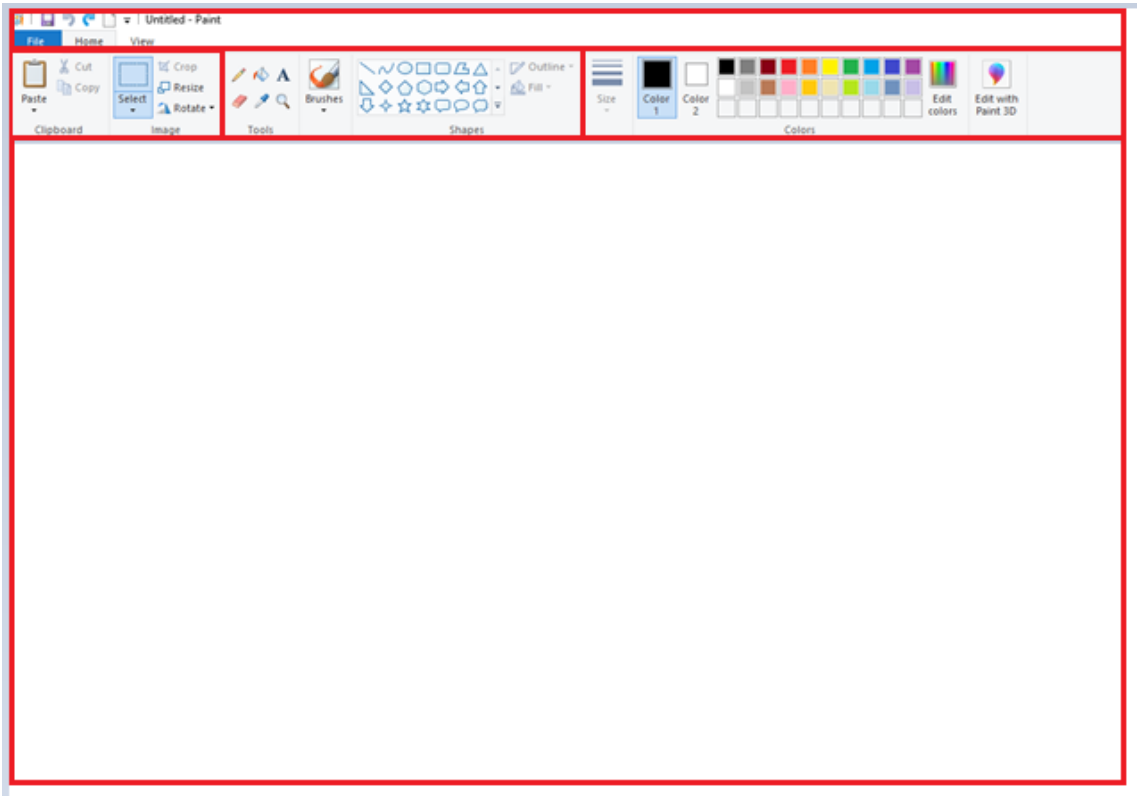



### 2. Paint

In this exercise, you will clone the interface of the Paint application.



Hint: You can break it down into smaller components to implement.




Details of the components: