Advanced CSS Assignment Solution

1.Explain the purpose of using the var() function in css. Also you have created two buttons with id named

primaryBtn and secondaryBtn which should be given background colors using the var() function. The color

code for primaryColor is #00b7ff and secondaryColor is #6c757d.

The var() function in CSS is used to implement custom properties, often referred to as CSS variables. It allows you to store values in a variable and reuse them throughout your CSS, making your code more maintainable, flexible, and easier to update.

Purpose of var() in CSS:

Reusability: You can define a value once and reuse it across multiple selectors, reducing repetition.

Maintainability: If you need to update a value (like a color or size), you can do it in one place rather than searching through your entire stylesheet.

Dynamic Changes: CSS variables can be dynamically changed with JavaScript, allowing for real-time updates of styles.

CODE

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>CSS Variables</title>

</head>

<style>

:root {

--primaryColor: #00b7ff;

--secondaryColor: #6c757d;

}

button {

font-size: 25px;

border: none;

padding: 5px 20px;

color: white;

}

#primaryBtn {

background-color: var(--primaryColor);

}

#secondaryBtn {

background-color: var(--secondaryColor);

}

</style>

<body>

<button id="primaryBtn">Primary Button</button>

<button id="secondaryBtn">Secondary Button</button>

</body>

</html>

2 Create a 3D cube using the transform property of CSS.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Cube</title>

</head>

<style>

body {

display: flex;

justify-content: center;

align-items: center;

min-height: 100vh;

margin: 0;

background-color: #f0f0f0;

}

.scene {

perspective: 1000px;

}

.cube {

width: 200px;

height: 200px;

position: relative;

transform-style: preserve-3d;

transform: rotateX(20deg) rotateY(30deg);

}

.face {

position: absolute;

width: 200px;

height: 200px;

background-color: rgba(0, 123, 255, 0.7);

border: 1px solid #666;

display: flex;

justify-content: center;

align-items: center;

font-size: 20px;

font-weight: bold;

opacity: 0.8;

}

.front {

transform: translateZ(100px);

}

.back {

transform: rotateY(180deg) translateZ(100px);

}

.right {

transform: rotateY(90deg) translateZ(100px);

}

.left {

transform: rotateY(-90deg) translateZ(100px);

}

.top {

transform: rotateX(90deg) translateZ(100px);

}

.bottom {

transform: rotateX(-90deg) translateZ(100px);

}

</style>

<body>

<div class="scene">

<div class="cube">

<div class="face front">Front</div>

<div class="face back">Back</div>

<div class="face right">Right</div>

<div class="face left">Left</div>

<div class="face top">Top</div>

<div class="face bottom">Bottom</div>

</div>

</div>

</body>

</html>

3.Create a simple circular loader which will rotate continuously to look like a loading screen on a website.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Circular Loader</title>

</head>

<style>

body {

display: flex;

justify-content: center;

align-items: center;

min-height: 100vh;

margin: 0;

background-color: #f0f0f0;

}

.loader {

border: 4px solid rgba(0, 0, 0, 0.1);

border-left-color: #007bff;

border-radius: 50%;

width: 40px;

height: 40px;

animation: spin 1s linear infinite;

}

@keyframes spin {

0% {

transform: rotate(0deg);

}

100% {

transform: rotate(360deg);

}

}

</style>

<body>

<div class="loader"></div>

</body>

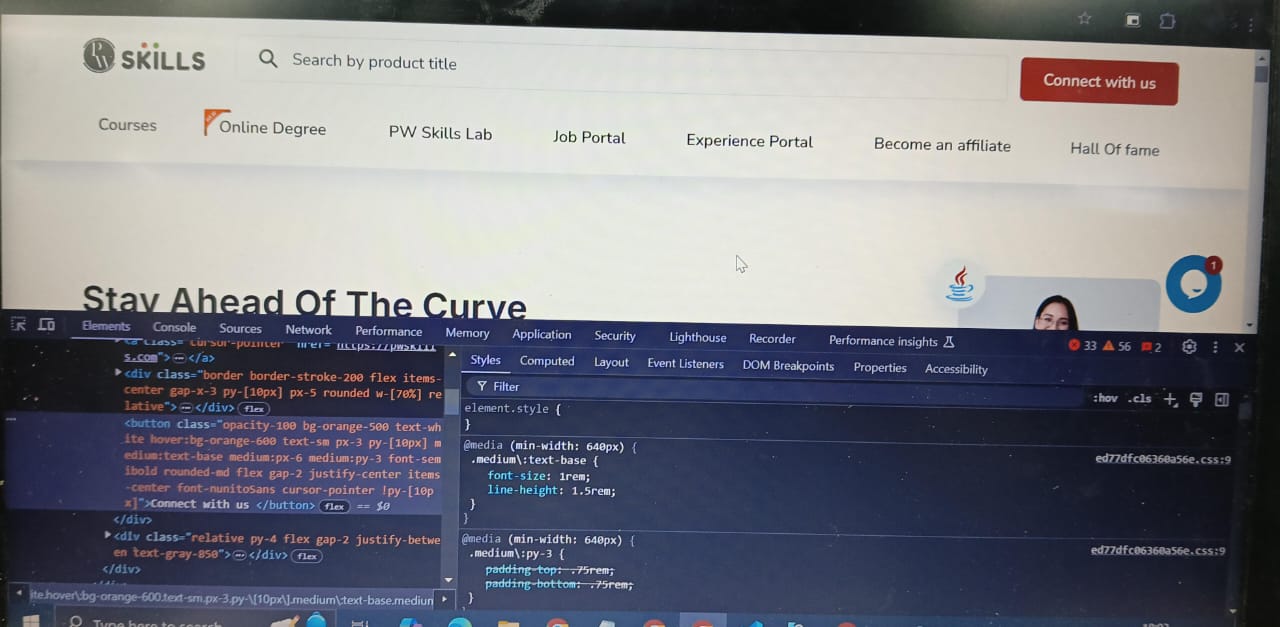
</html>

4.solution

A screenshot of a computer

Description automatically generated

5.SOLUTION



A screenshot of a computer

Description automatically generated