**CSS Transitions and Animations**

CSS transitions and animations can be used to create smooth and engaging visual effects on your web page.

Let's cover both concepts and see some examples.

**CSS Transitions**

CSS transitions allow you to change property values smoothly (over a given duration).

**Basic Syntax**

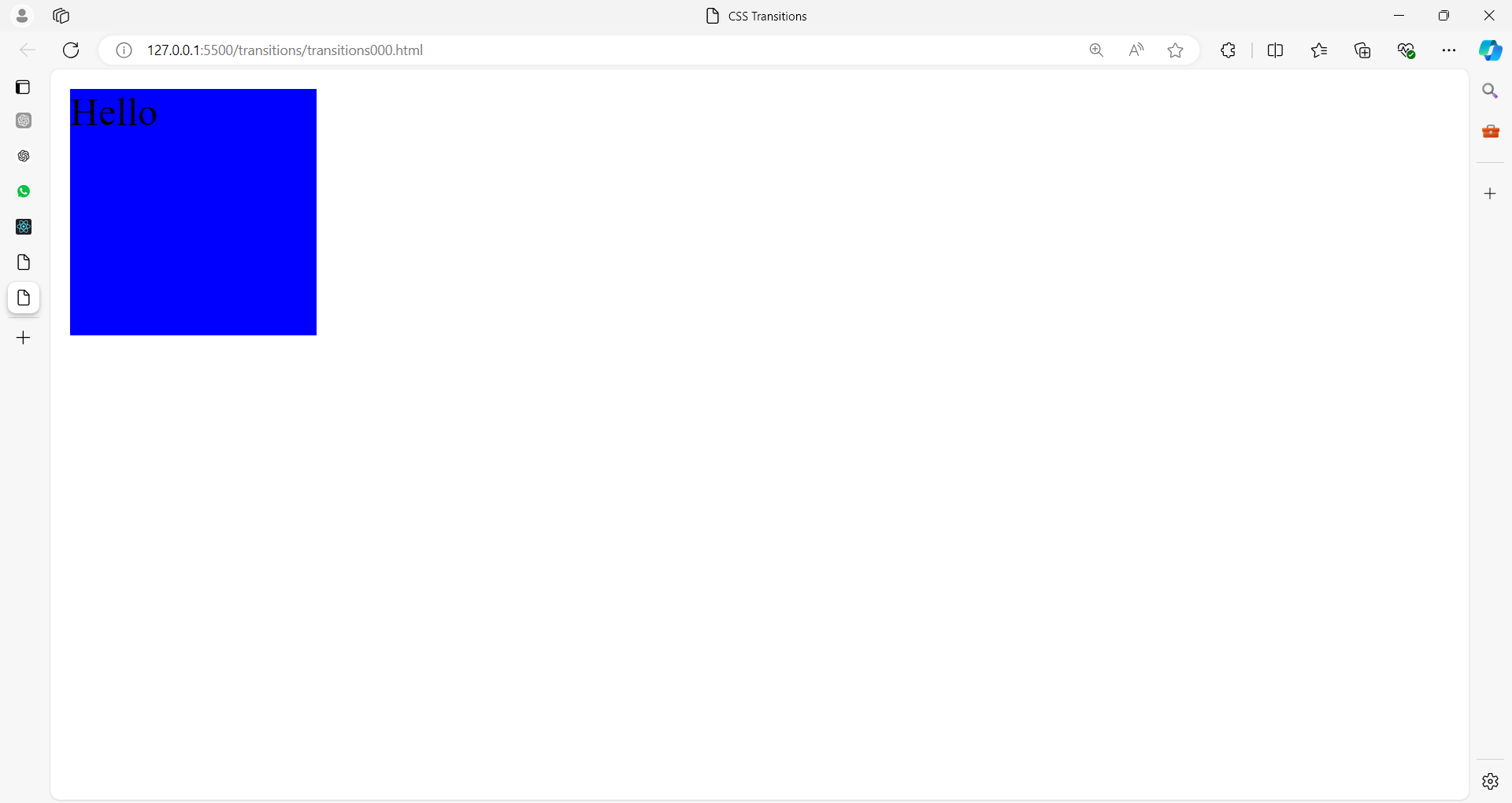
selector {

**transition: property duration timing-function delay;**

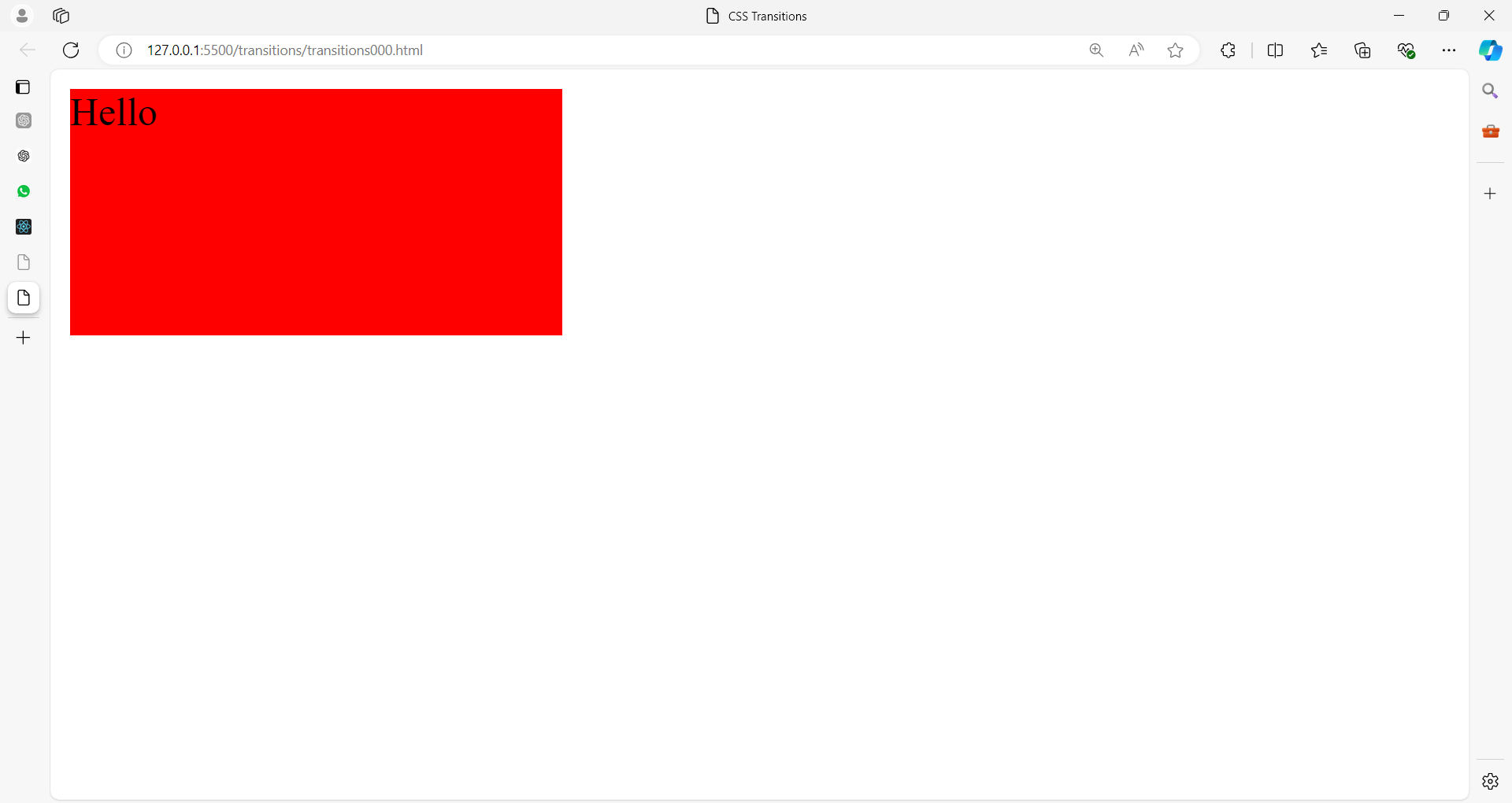
}

* **property:** The CSS property you want to apply the transition to (e.g., width, height, background-color).
* **duration:** The time it takes for the transition to complete (e.g., 2s for 2 seconds, 500ms for 500 milliseconds).
* **timing-function:** The speed curve of the transition effect (e.g., ease, linear, ease-in, ease-out, cubic-bezier).
* **delay:** The time to wait before starting the transition (e.g., 1s for 1 second, 0s for no delay).

**Example**



**After hovering on the box**



<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>CSS Transitions</title>

<style>

.box {

width: 100px;

height: 100px;

background-color: blue;

**transition: background-color 1s, width 2s;**

}

.box:hover {

background-color: red;

width: 200px;

}

</style>

</head>

<body>

<div class="box">Hello</div>

</body>

</html>

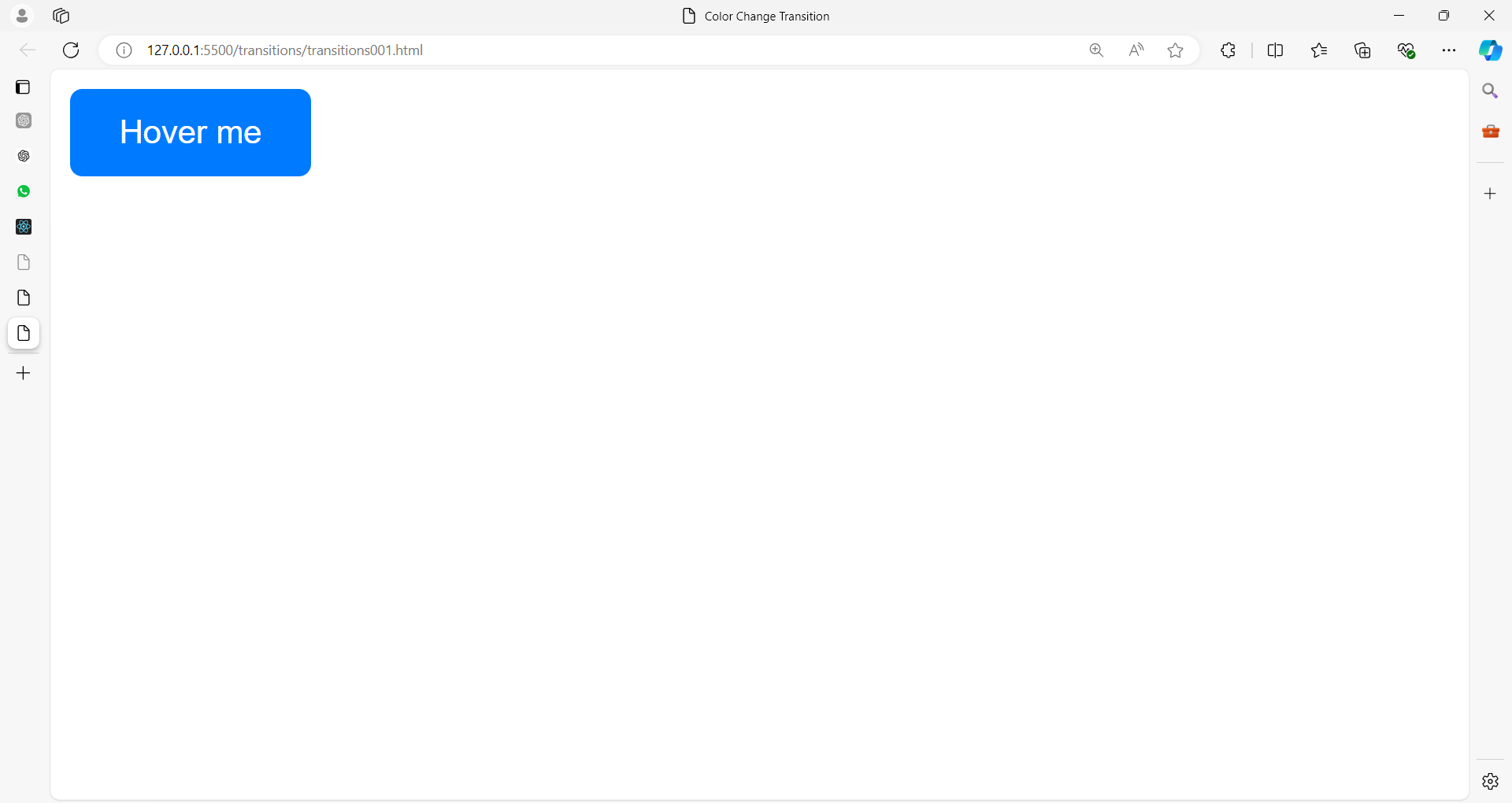
In this example:

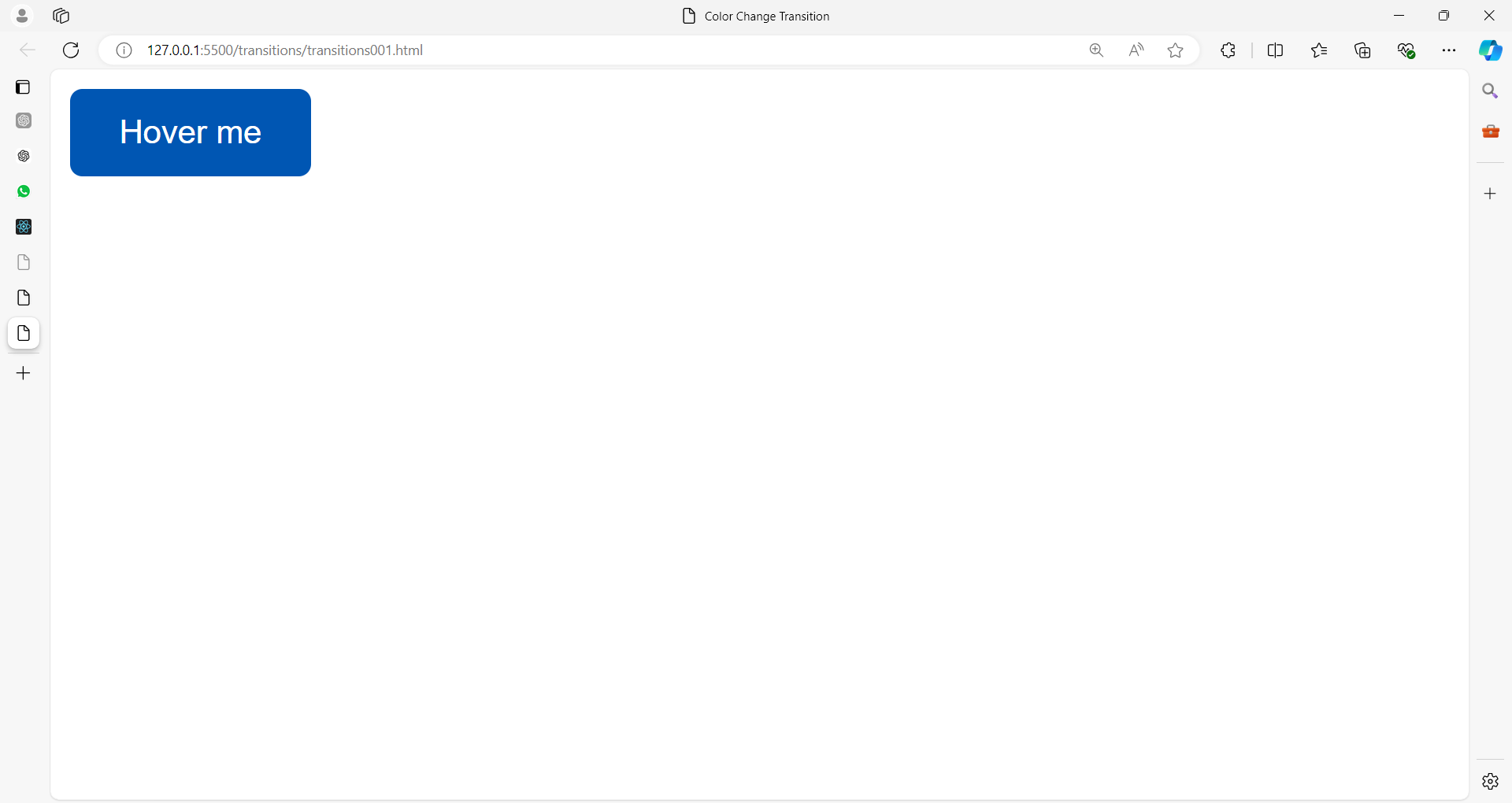
* The box class has a **transition** applied to **background-color** and **width**.
* When you **hover** over the .box, the background color changes to red over 1 second, and the width changes to 200px over 2 seconds.

### **Examples**

### **1. Simple Color Change**

This example changes the background color of a button when hovered.





<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Color Change Transition</title>

<style>

.button {

background-color: #007bff;

color: white;

padding: 10px 20px;

border: none;

border-radius: 5px;

cursor: pointer;

**transition: background-color 0.5s;**

}

.button:hover {

background-color: #0056b3;

}

</style>

</head>

<body>

<button class="button">Hover me</button>

</body>

</html>

Let’s break down the CSS3 used in the provided program line by line:

**.button {**

**background-color: #007bff;**

**color: white;**

**padding: 10px 20px;**

**border: none;**

**border-radius: 5px;**

**cursor: pointer;**

**transition: background-color 0.5s;**

**}**

1. **.button { ... }**: This selector targets HTML elements with the class button, applying the enclosed styles to those elements.
2. **background-color: #007bff;**: Sets the background color of the button to a shade of blue. The color code #007bff is a specific hex value representing this color.
3. **color: white;**: Sets the text color inside the button to white, ensuring good contrast against the blue background.
4. **padding: 10px 20px;**: Adds space inside the button element. 10px is applied to the top and bottom, and 20px is applied to the left and right, creating a balanced, padded area around the button's text.
5. **border: none;**: Removes any default border around the button. By default, buttons might have a border, and this line ensures that there’s none.
6. **border-radius: 5px;**: Rounds the corners of the button by 5px. This gives the button a smooth, rounded appearance.
7. **cursor: pointer;**: Changes the mouse cursor to a pointer (usually a hand icon) when hovering over the button, indicating that the button is clickable.
8. **transition: background-color 0.5s;**: Specifies a transition effect for the background color property. When the background color changes (for example, during a hover state), it will smoothly transition over 0.5 seconds. This creates a fading effect between the original color and the color set for the hover state.

**.button:hover {**

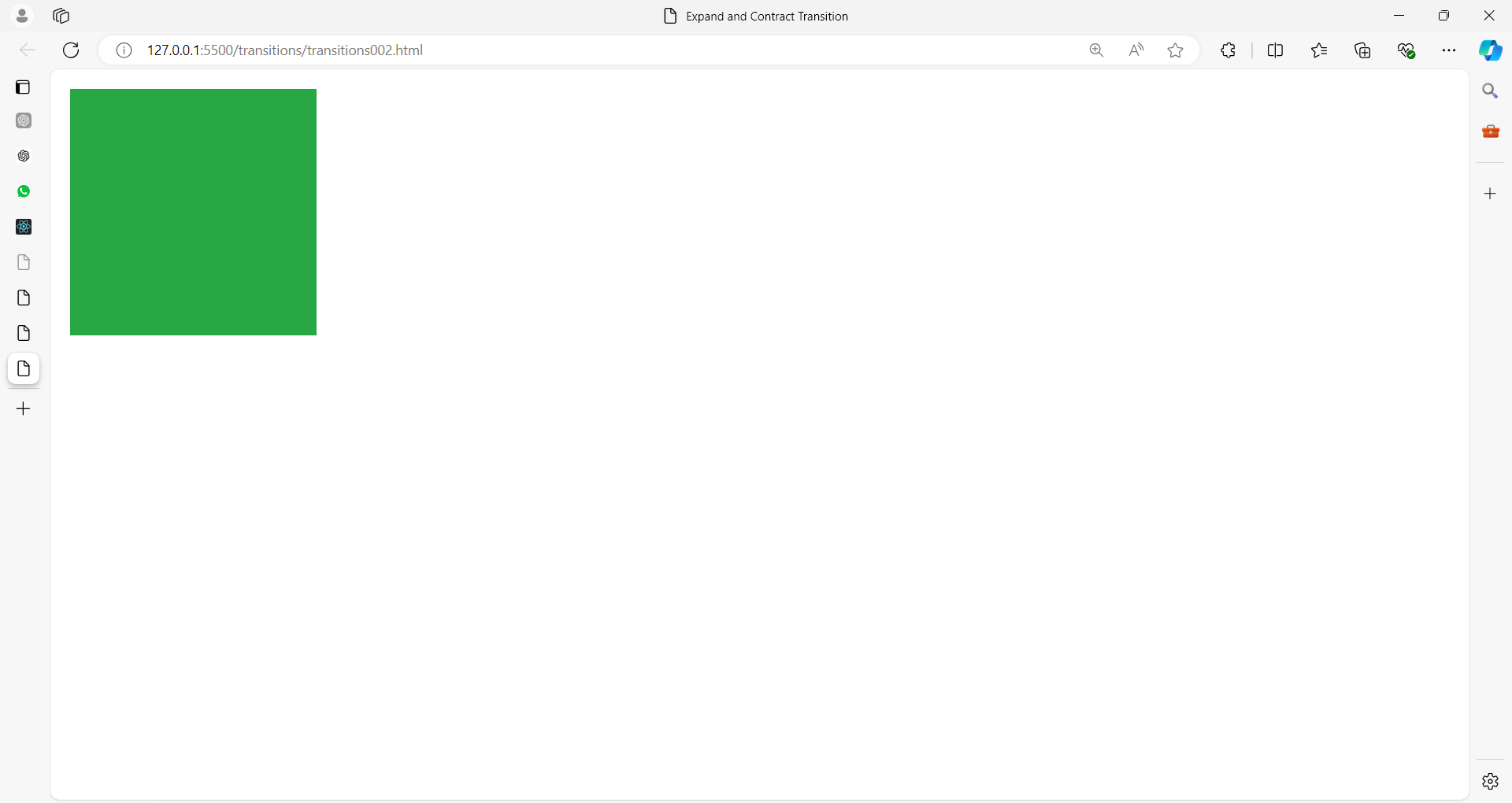
**background-color: #0056b3;**

**}**

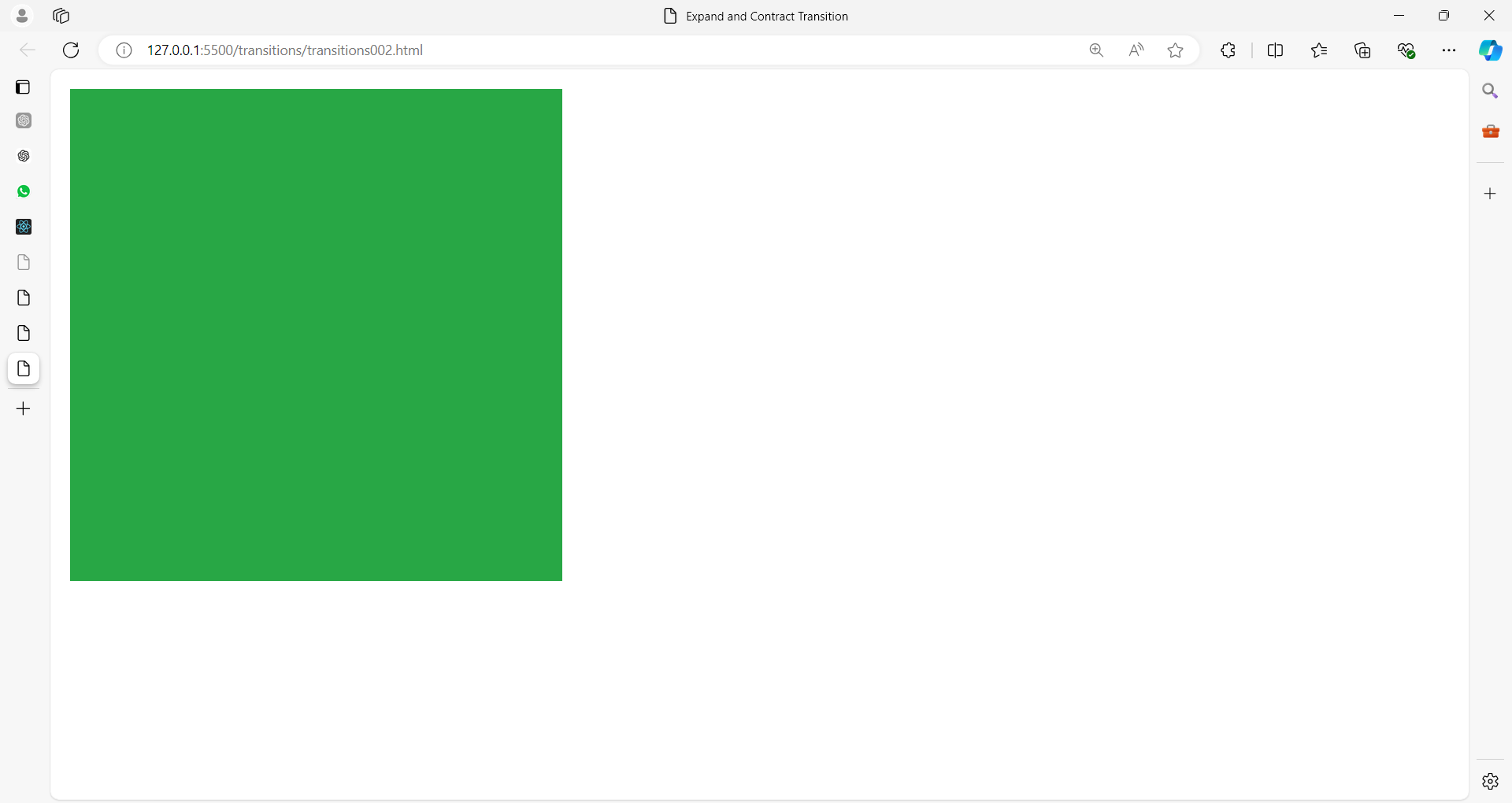
1. **.button:hover { ... }**: This selector applies styles when the button is being hovered over by the mouse. The :hover pseudo-class is used to define these styles.
2. **background-color: #0056b3;**: Changes the background color of the button to a darker shade of blue when the button is hovered over. The color code #0056b3 represents this darker shade, providing a visual feedback effect to the user.

### **2. Expand and Contract**

This example enlarges a box when hovered.



After getting cursor on the pic



<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Expand and Contract Transition</title>

<style>

.box {

width: 100px;

height: 100px;

background-color: #28a745;

**transition: width 0.5s, height 0.5s;**

}

.box:hover {

width: 200px;

height: 200px;

}

</style>

</head>

<body>

<div class="box"></div>

</body>

</html>

Let’s break down the CSS3 used in this example:

**.box {**

**width: 100px;**

**height: 100px;**

**background-color: #28a745;**

**transition: width 0.5s, height 0.5s;**

**}**

1. **.box { ... }**: This selector targets HTML elements with the class box and applies the enclosed styles to them.
2. **width: 100px;**: Sets the initial width of the box to 100px.
3. **height: 100px;**: Sets the initial height of the box to 100px.
4. **background-color: #28a745;**: Applies a green background color to the box. The color code #28a745 represents this shade of green.
5. **transition: width 0.5s, height 0.5s;**: Specifies the transition effects for the width and height properties. When these properties change, the transition will occur over 0.5 seconds. This creates a smooth animation effect as the box’s dimensions change.

**.box:hover {**

**width: 200px;**

**height: 200px;**

**}**

1. **.box:hover { ... }**: This selector applies styles when the .box element is hovered over by the mouse. The :hover pseudo-class is used to define these styles.
2. **width: 200px;**: Changes the width of the box to 200px when hovered over.
3. **height: 200px;**: Changes the height of the box to 200px when hovered over.

**How It Works:**

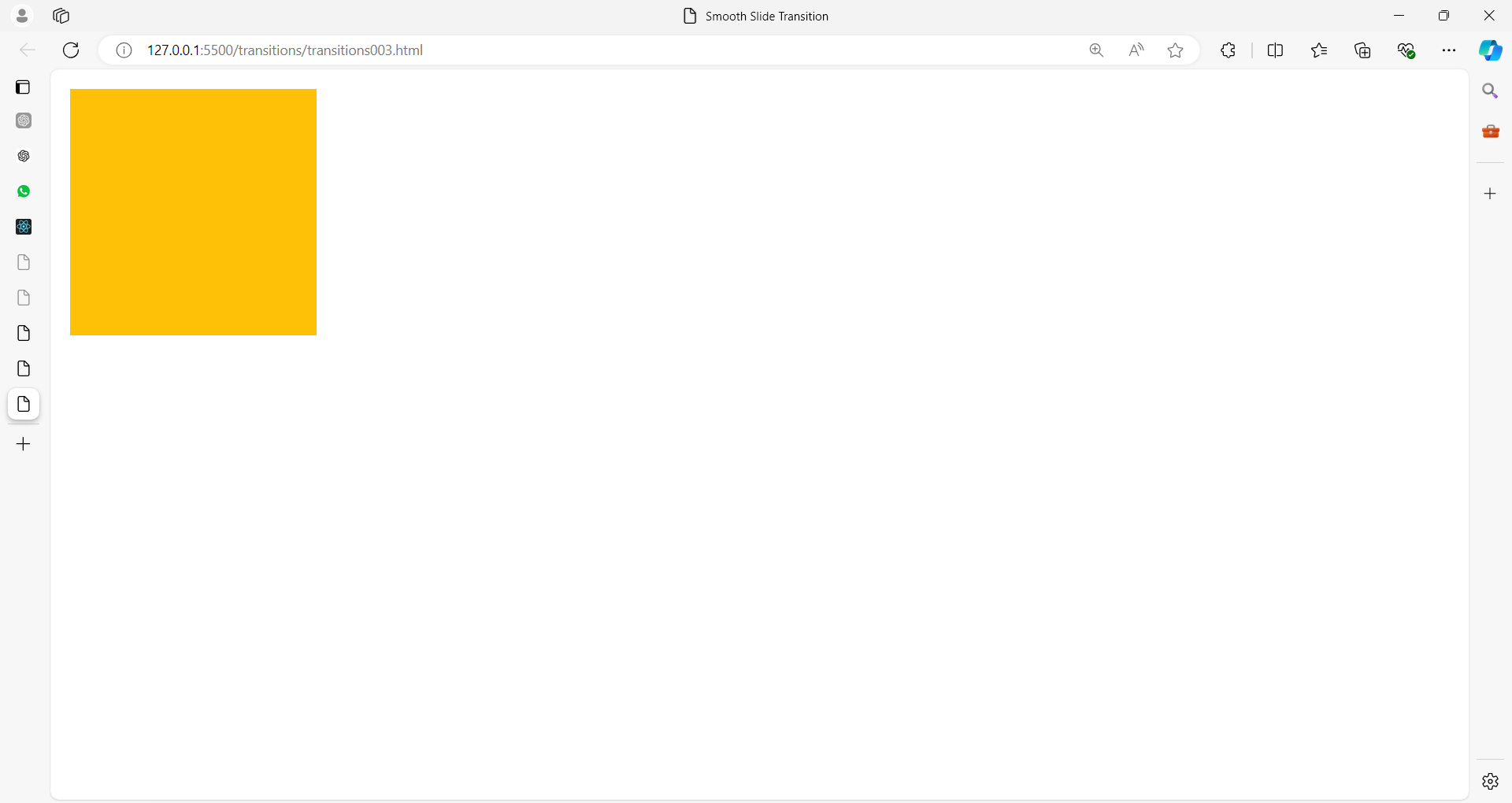
* **Initial State**: The .box element starts with a width and height of 100px and a green background.
* **Hover State**: When the user hovers over the box, its width and height expand to 200px each.
* **Transition Effect**: The change in width and height occurs over 0.5 seconds, creating a smooth expansion and contraction effect.

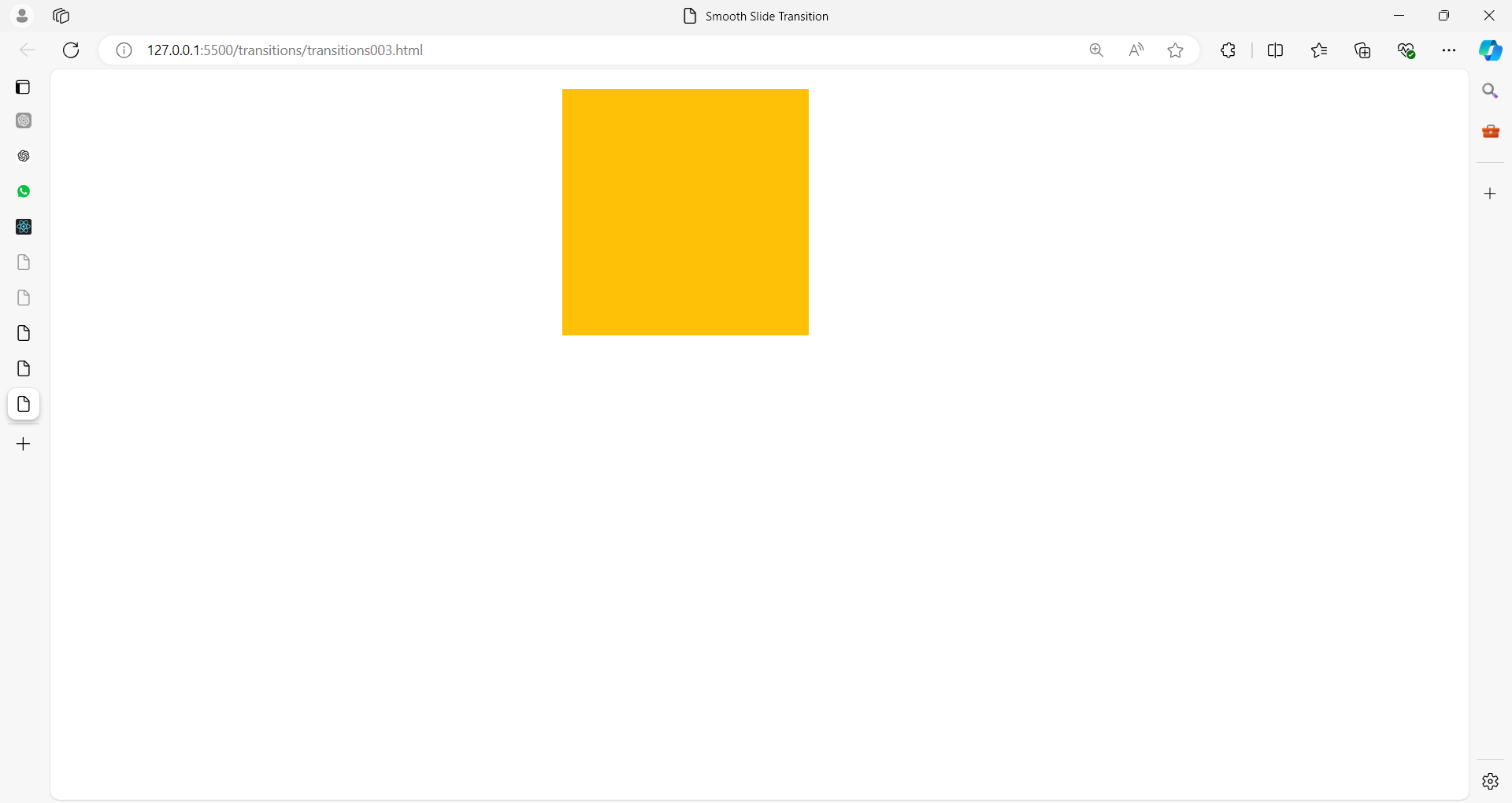
**Visual Effect:**

The box will smoothly grow in size when hovered over and smoothly return to its original size when the hover ends. This visual feedback can be useful for interactive elements on a web page.

### **3. Smooth Slide**

This example moves a box to the right when hovered.





### **<!DOCTYPE html>**

### **<html lang="en">**

### **<head>**

### **<meta charset="UTF-8">**

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

### **<title>Smooth Slide Transition</title>**

### **<style>**

### **.box {**

### **width: 100px;**

### **height: 100px;**

### **background-color: #ffc107;**

### **transition: transform 0.5s;**

### **}**

### **.box:hover {**

### **transform: translate(200px);**

/\*transform: translateX(200px);

            transform: translateY(200px);\*/

### **}**

### **</style>**

### **</head>**

### **<body>**

### **<div class=”box”</div>**

### **</body>**

### **</html>**

Here’s a line-by-line explanation of the CSS3 used in this example:

**.box {**

**width: 100px;**

**height: 100px;**

**background-color: #ffc107;**

**transition: transform 0.5s;**

**}**

1. **.box { … }**: This selector applies styles to the HTML elements with the class box.
2. **width: 100px;**: Sets the initial width of the box to 100px.
3. **height: 100px;**: Sets the initial height of the box to 100px.
4. **background-color: #ffc107;**: Applies a yellow background color to the box. The color code #ffc107 represents this shade of yellow.
5. **transition: transform 0.5s;**: Specifies that any changes to the transform property will animate over 0.5 seconds. The transition property is used to create smooth animations for changes in CSS properties.

**.box:hover {**

**transform: translate(200px);**

**}**

1. **.box:hover { … }**: This selector applies styles when the .box element is hovered over by the mouse. The :hover pseudo-class is used to define these styles.
2. **transform: translate(200px);**: Applies a transformation that moves the box horizontally by 200px. The translate function shifts the element along the X-axis.

**How It Works:**

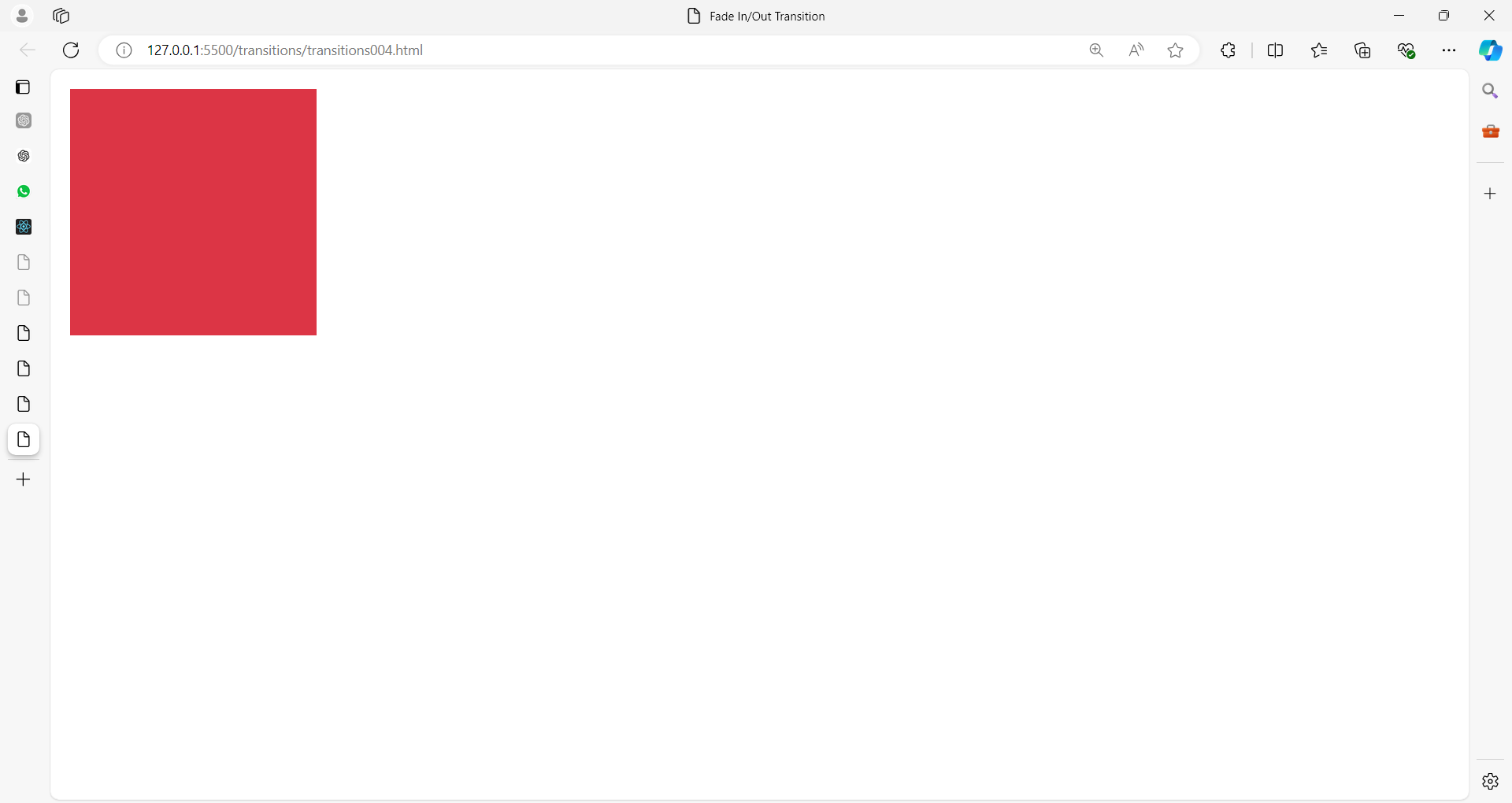
* **Initial State**: The .box element starts with a width and height of 100px and a yellow background color. It is positioned at its original location.
* **Hover State**: When the user hovers over the box, the transform property causes the box to slide 200px to the right.
* **Transition Effect**: The sliding movement occurs smoothly over 0.5 seconds, creating a visually appealing sliding effect.

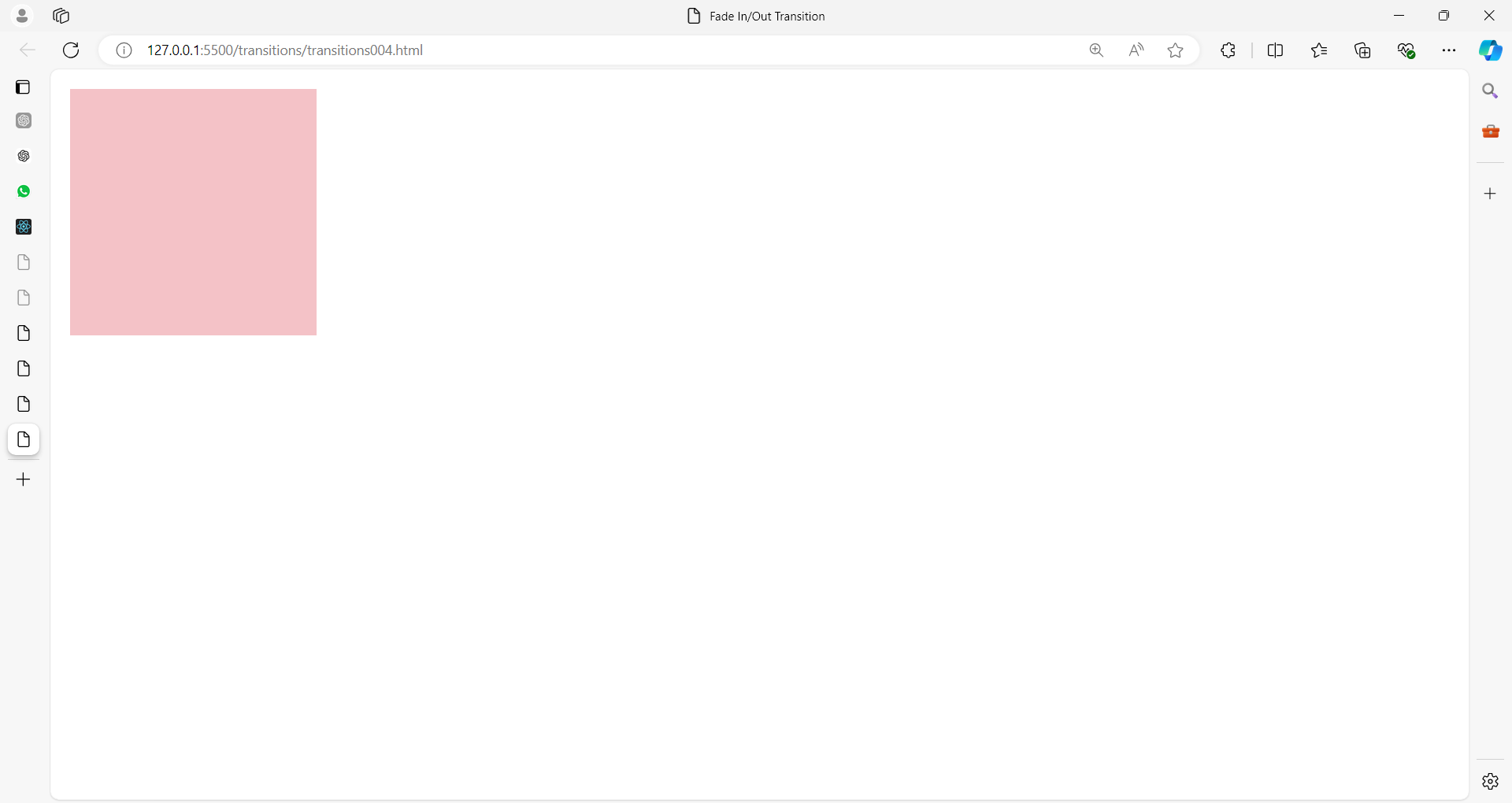
**Visual Effect:**

The box will slide smoothly to the right by 200px when hovered over and return to its original position when the hover ends. This effect can be useful for interactive UI elements, such as buttons or menu items, to attract attention or provide visual feedback.

### **Fade In/Out**

This example changes the opacity of a box to create a fade effect.





<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Fade In/Out Transition</title>

<style>

.box {

width: 100px;

height: 100px;

background-color: #dc3545;

**transition: opacity 0.5s;**

}

.box:hover {

opacity: 0.3;

}

</style>

</head>

<body>

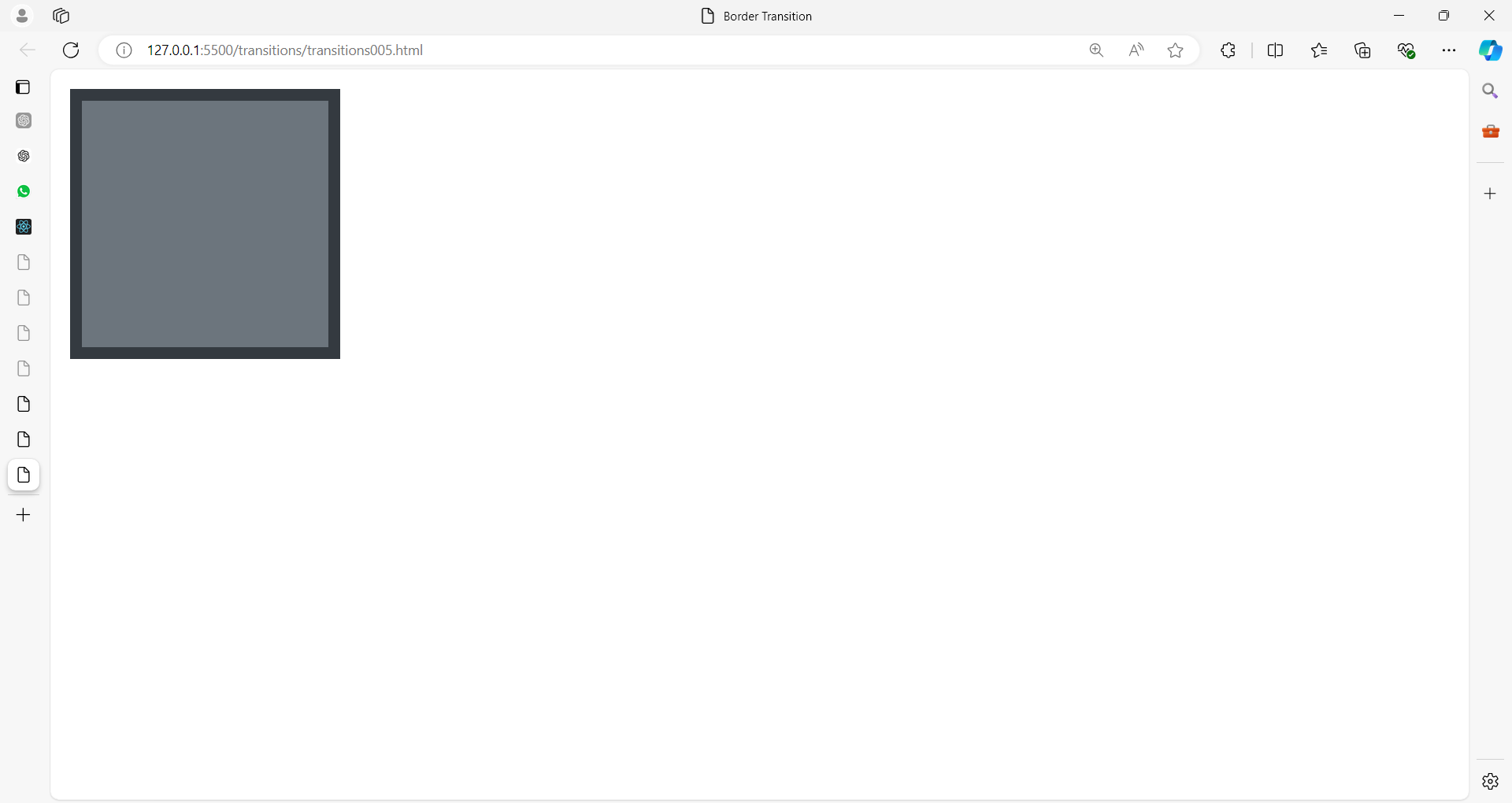
<div class="box"></div>

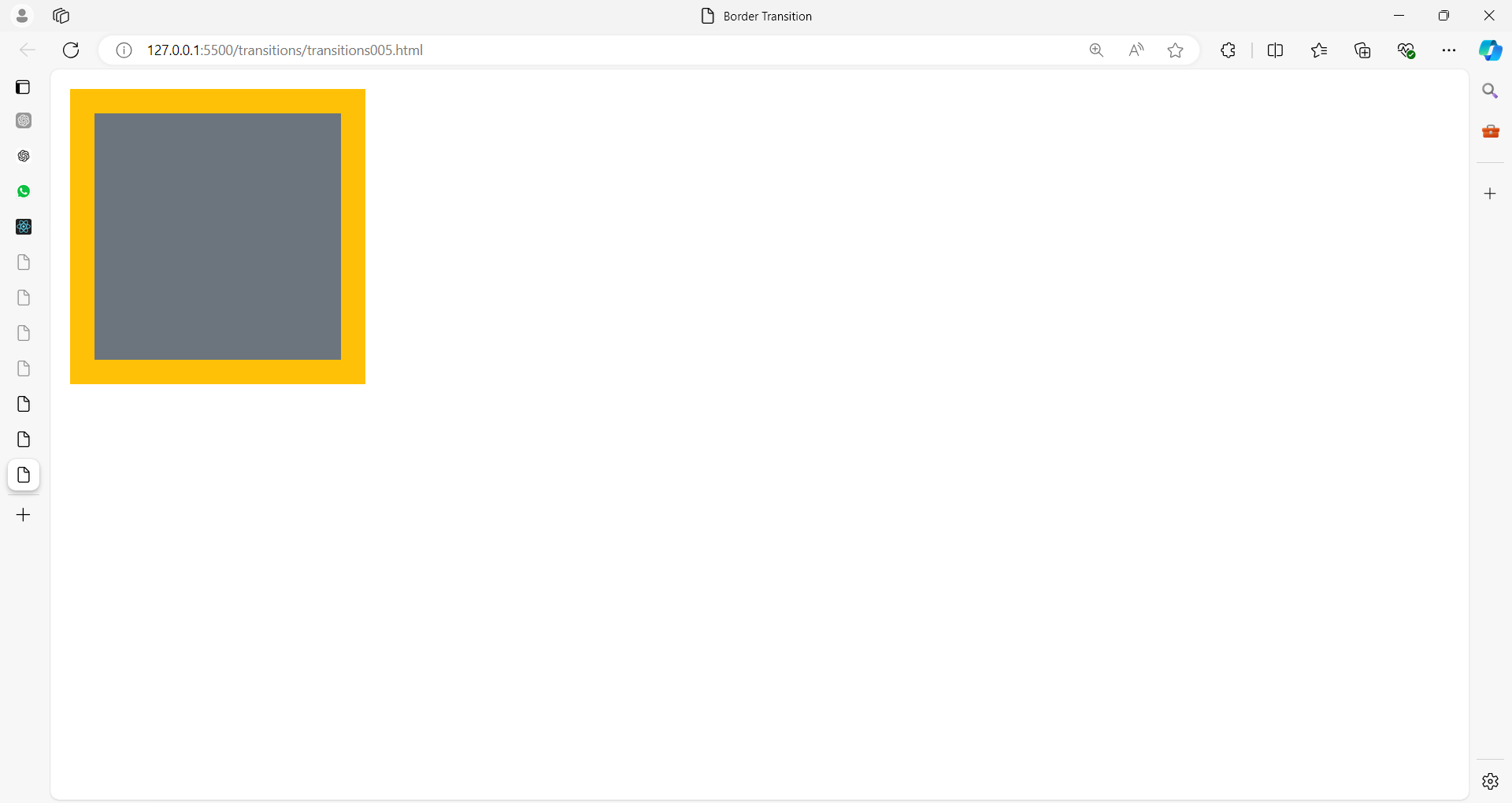
</body>

</html>

### **5. Border Transition**

This example changes the border width and color of a box when hovered.





<!DOCTYPE html>

<html lang=”en”>

<head>

<meta charset=”UTF-8”>

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

<title>Border Transition</title>

<style>

.box {

width: 100px;

height: 100px;

background-color: #6c757d;

border: 5px solid #343a40;

**transition: border 0.5s;**

}

.box:hover {

border: 10px solid #ffc107;

}

</style>

</head>

<body>

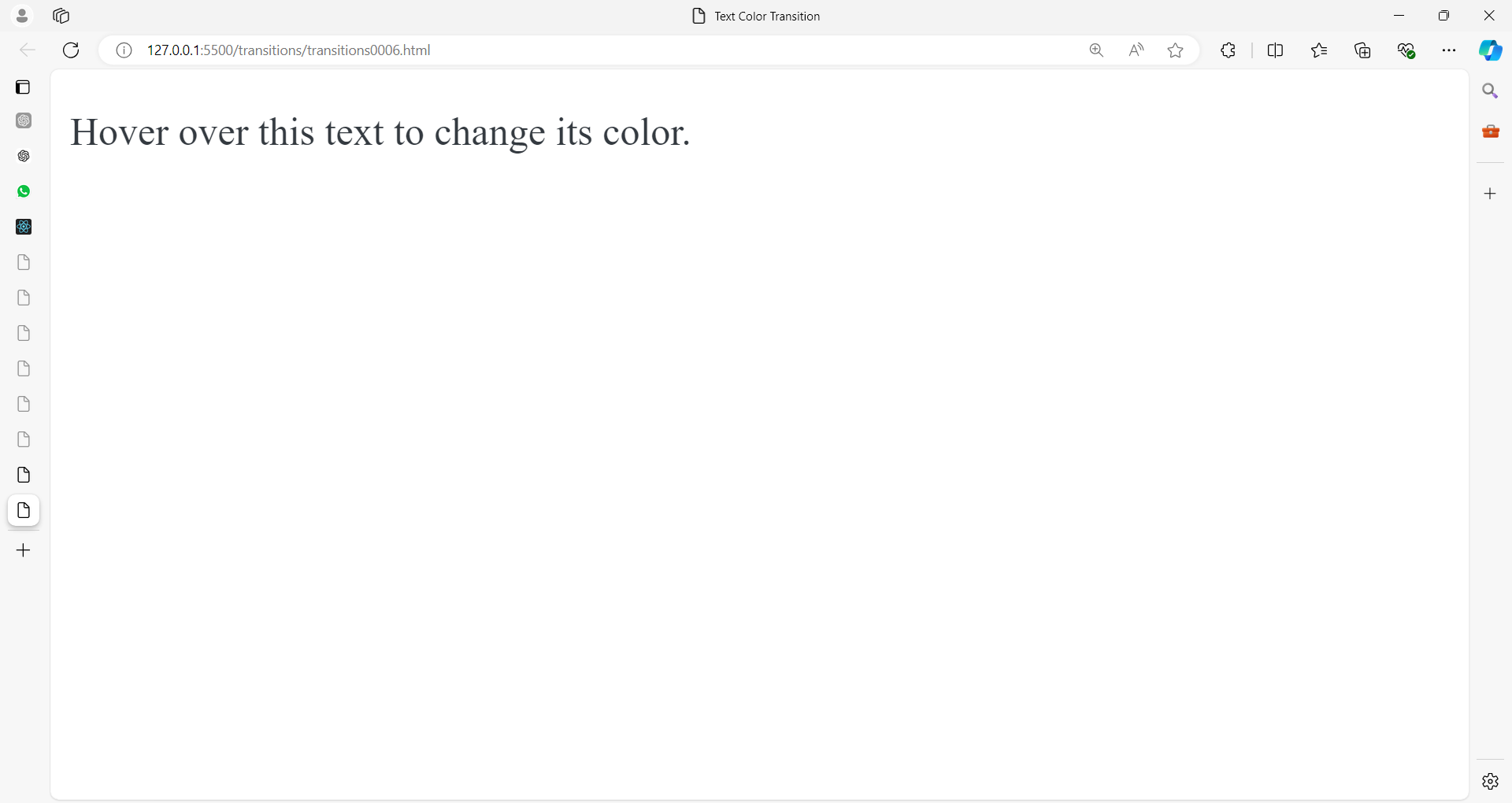
<div class=”box”></div>

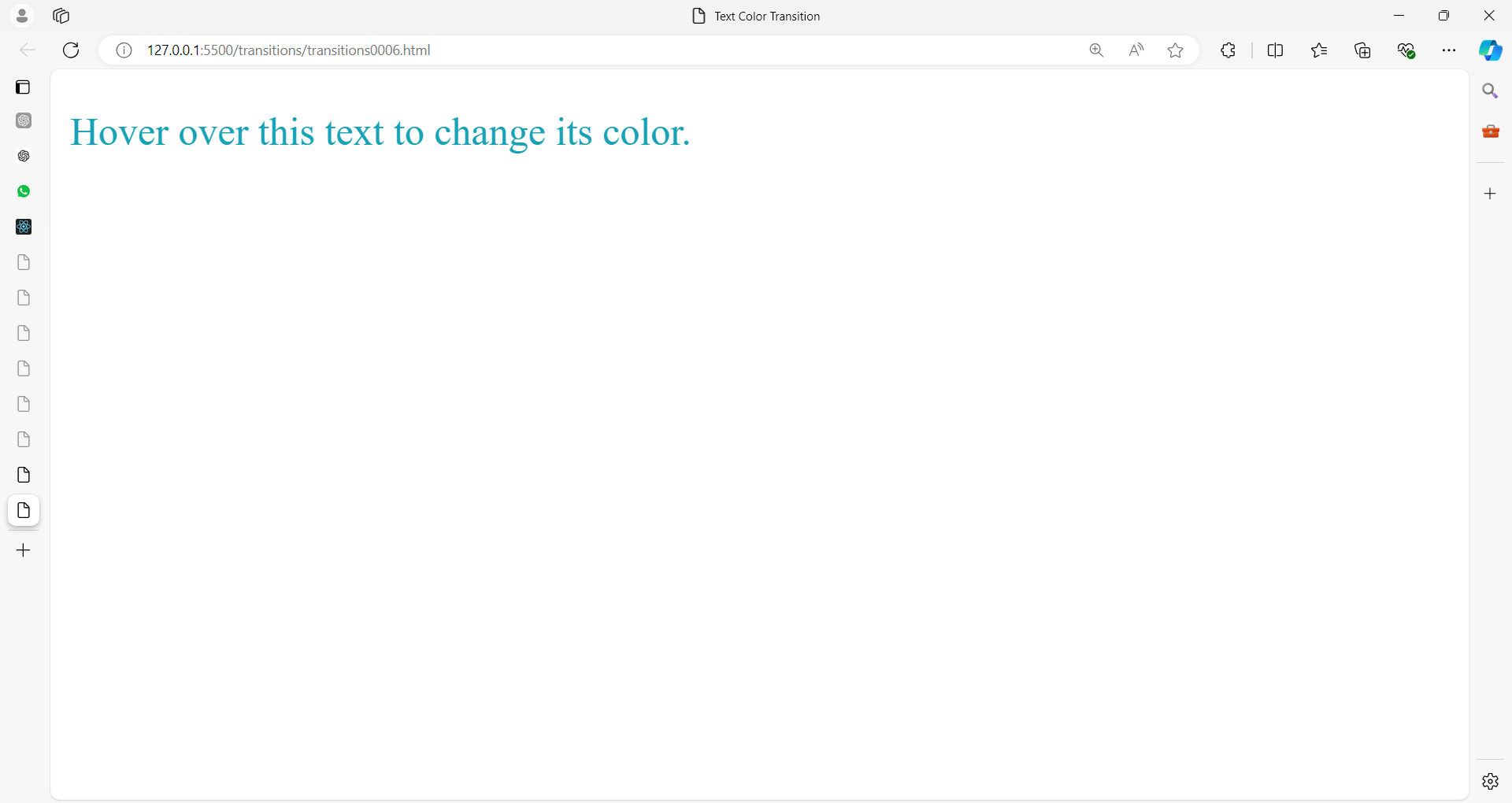
</body>

</html>

### **Text Color Transition**

This example changes the text color of a paragraph when hovered.





<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Text Color Transition</title>

<style>

.text {

color: #343a40;

**transition: color 0.5s;**

}

.text:hover {

color: #17a2b8;

}

</style>

</head>

<body>

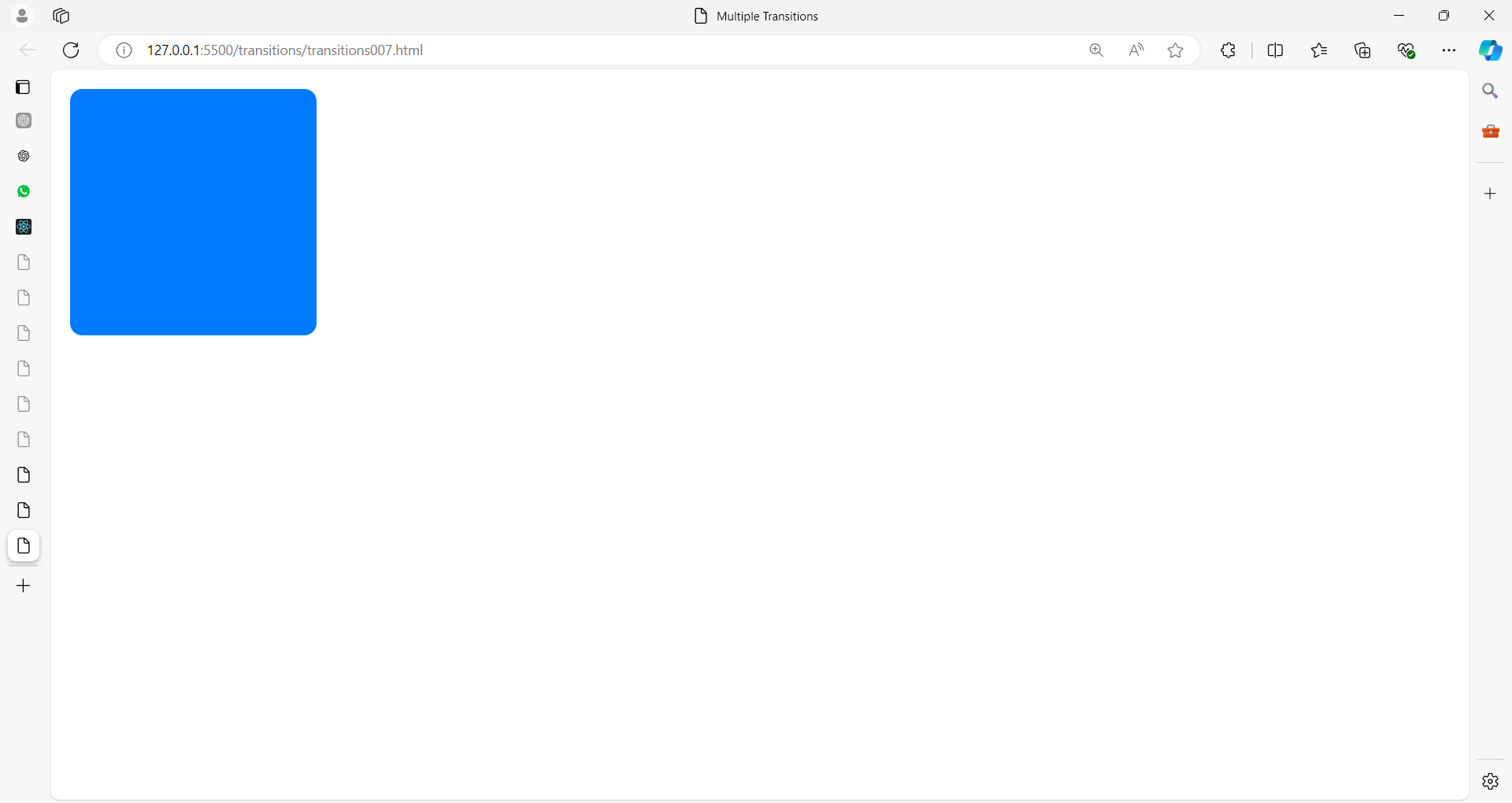
<p class="text">Hover over this text to change its color.</p>

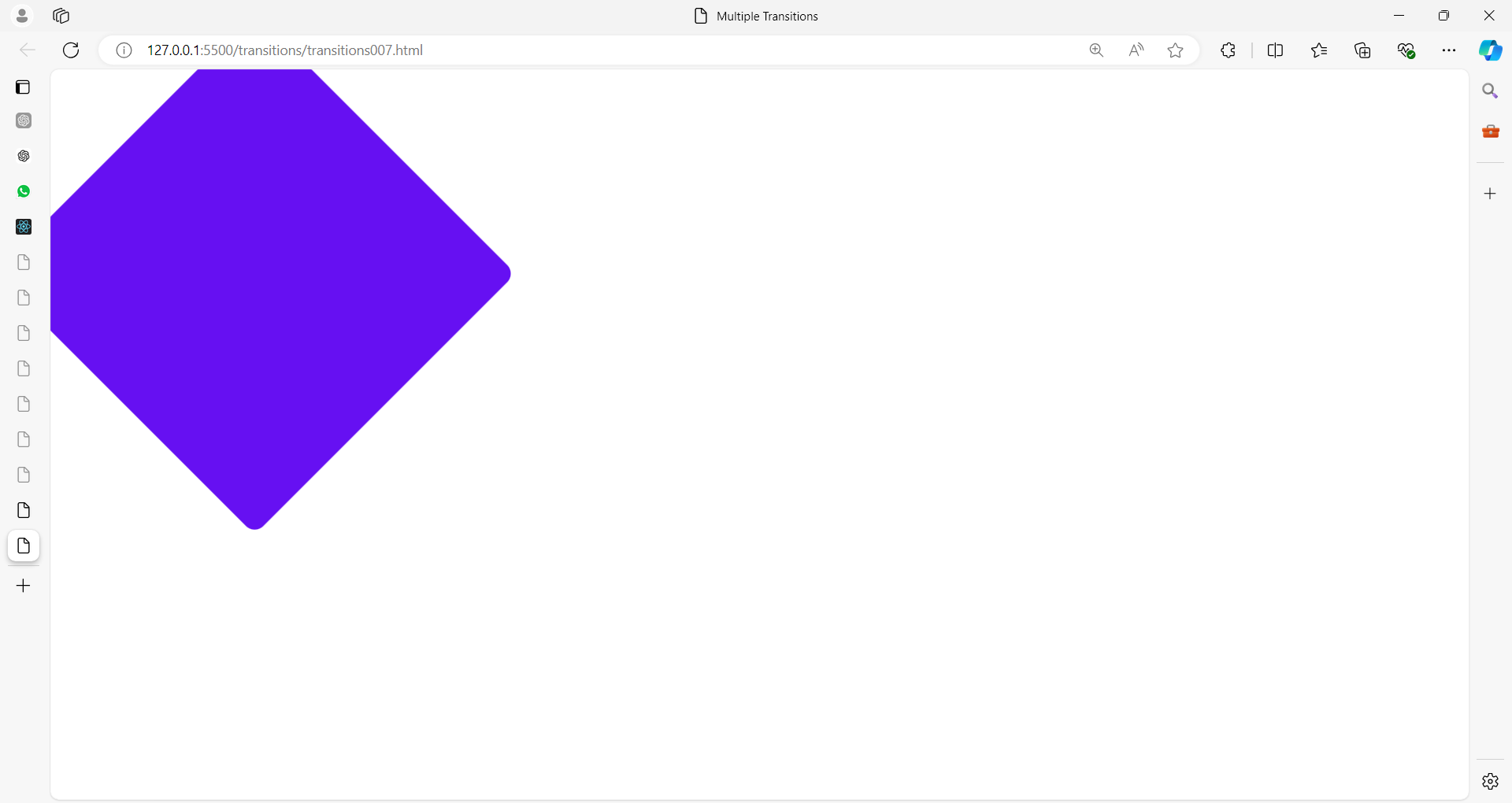
</body>

</html>

### **7. Multiple Transitions**

This example demonstrates the use of multiple transitions on a single element.





<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Multiple Transitions</title>

<style>

.box {

width: 100px;

height: 100px;

background-color: #007bff;

border-radius: 5px;

**transition: background-color 0.5s, width 0.5s, height 0.5s, transform 0.5s;**

}

.box:hover {

background-color: #6610f2;

width: 150px;

height: 150px;

**transform: rotate(45deg);**

}

</style>

</head>

<body>

<div class="box"></div>

</body>

</html>

### **8. Width Transition with Flexbox**

This example uses flexbox and transitions to animate the width of an item.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Flexbox Width Transition</title>

<style>

.container {

display: flex;

width: 100%;

height: 100px;

background-color: #f8f9fa;

}

.item {

flex: 1;

background-color: #007bff;

**transition: flex 0.5s;**

}

.item:hover {

flex: 3;

background-color: #d20e25;

}

</style>

</head>

<body>

<div class="container">

<div class="item"></div>

<div class="item"></div>

<div class="item"></div>

</div>

</body>

</html>

**CSS Animations**

CSS animations allow you to animate transitions between multiple CSS styles.

You can define keyframes for the start and end states of the animation, as well as any intermediate waypoints.

**Basic Syntax**

1. **Define keyframes using @keyframes.**

@keyframes animation-name {

from { /\* styles \*/ }

to { /\* styles \*/ }

}

/\* or \*/

@keyframes animation-name {

0% { /\* styles \*/ }

100% { /\* styles \*/ }

/\* you can also add intermediate steps: 25%, 50%, 75%, etc. \*/

}

1. **Apply the animation to an element.**

**animation**: animation-name duration timing-function delay iteration-count direction fill-mode;

}

* animation-name: The name of the keyframes defined.
* duration: The duration of the animation.
* timing-function: The speed curve of the animation.
* delay: The delay before the animation starts.
* iteration-count: The number of times the animation should repeat (e.g., infinite for infinite repetitions).
* direction: The direction of the animation (e.g., normal, reverse, alternate).
* fill-mode: Specifies how a CSS animation should apply styles to its target before and after it is executing (e.g., none, forwards, backwards, both).

**Example**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>CSS Animations</title>

<style>

@keyframes slide {

from {

**transform: translateX(0);**

}

to {

**transform: translateX(200px);**

}

}

.box {

width: 100px;

height: 100px;

background-color: blue;

**animation: slide 2s ease-in-out infinite alternate;**

}

</style>

</head>

<body>

<div class="box"></div>

</body>

</html>

In this example:

* The slide animation moves the element from translateX(0) to translateX(200px).
* The .box class applies the slide animation, which takes 2 seconds, uses an ease-in-out timing function, repeats infinitely, and alternates direction each time.

**Examples**

### **1. Bouncing Ball Animation**

This example demonstrates a simple bouncing ball animation.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Bouncing Ball Animation</title>

<style>

.ball {

width: 50px;

height: 50px;

background-color: #007bff;

border-radius: 50%;

position: absolute;

top: 0;

**animation: bounce 2s infinite;**

}

@keyframes bounce {

0%, 100% {

top: 0;

**animation-timing-function: ease-in;**

}

50% {

top: 300px;

**animation-timing-function: ease-out;**

}

}

</style>

</head>

<body>

<div class="ball"></div>

</body>

</html>

### **2. Rotating Square Animation**

This example shows a square that rotates continuously.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Rotating Square Animation</title>

<style>

.square {

width: 100px;

height: 100px;

background-color: #28a745;

**animation: rotate 3s linear infinite;**

}

@keyframes rotate {

from {

transform: rotate(0deg);

}

to {

transform: rotate(360deg);

}

}

</style>

</head>

<body>

<div class="square"></div>

</body>

</html>

### **3. Fade In/Out Animation**

This example makes an element fade in and out.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Fade In/Out Animation</title>

<style>

.box {

width: 100px;

height: 100px;

background-color: #dc3545;

**animation: fade 2s infinite;**

}

@keyframes fade {

0%, 100% {

opacity: 0;

}

50% {

opacity: 1;

}

}

</style>

</head>

<body>

<div class="box"></div>

</body>

</html>

### **4. Sliding Text Animation**

This example slides text in from the left.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Sliding Text Animation</title>

<style>

.text {

font-size: 24px;

font-weight: bold;

position: relative;

**animation: slide 3s forwards;**

}

@keyframes slide {

from {

left: -100%;

}

to {

left: 0;

}

}

</style>

</head>

<body>

<div class="text">Hello, World!</div>

</body>

</html>

### **5. Pulsing Effect**

This example creates a pulsing effect on a button.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Pulsing Effect</title>

<style>

.button {

padding: 10px 20px;

background-color: #6f42c1;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

**animation: pulse 1s infinite;**

}

@keyframes pulse {

0%, 100% {

transform: scale(1);

}

50% {

transform: scale(1.1);

}

}

</style>

</head>

<body>

<button class="button">Pulse</button>

</body>

</html>

### **6. Moving Gradient Background**

This example animates a gradient background to create a moving effect.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Moving Gradient Background</title>

<style>

.background {

width: 100%;

height: 100vh;

background: linear-gradient(270deg, #ff5733, #33ff57, #3357ff);

background-size: 600% 600%;

**animation: gradient 15s ease infinite;**

}

@keyframes gradient {

0% {

background-position: 0% 50%;

}

50% {

background-position: 100% 50%;

}

100% {

background-position: 0% 50%;

}

}

</style>

</head>

<body>

<div class="background"></div>

</body>

</html>

### **7. Heartbeat Animation**

This example creates a heartbeat effect on a heart icon.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Heartbeat Animation</title>

<style>

.heart {

width: 100px;

height: 100px;

background-color: #e74c3c;

position: relative;

transform: rotate(-45deg);

**animation: heartbeat 1s infinite;**

}

.heart::before,

.heart::after {

content: "";

width: 100px;

height: 100px;

background-color: #e74c3c;

border-radius: 50%;

position: absolute;

}

.heart::before {

top: -50px;

left: 0;

}

.heart::after {

left: 50px;

top: 0;

}

@keyframes heartbeat {

0%, 100% {

transform: scale(1) rotate(-45deg);

}

50% {

transform: scale(1.2) rotate(-45deg);

}

}

</style>

</head>

<body>

<div class="heart"></div>

</body>

</html>

### **8. Typing Text Animation**

This example simulates a typing effect.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Typing Text Animation</title>

<style>

.typing-text {

font-size: 24px;

font-family: monospace;

white-space: nowrap;

overflow: hidden;

border-right: 2px solid;

**animation: typing 4s steps(20, end), blink-caret 0.75s step-end infinite;**

}

@keyframes typing {

from {

width: 0;

}

to {

width: 100%;

}

}

@keyframes blink-caret {

from, to {

border-color: transparent;

}

50% {

border-color: black;

}

}

</style>

</head>

<body>

<div class="typing-text">This is a typing animation effect.</div>

</body>

</html>

**Combining Transitions and Animations**

You can combine transitions and animations to create more complex effects.

Here’s an example that combines both:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Combined Transitions and Animations</title>

<style>

@keyframes bounce {

0%, 20%, 50%, 80%, 100% {

transform: translateY(0);

}

40% {

transform: translateY(-150px);

}

60% {

transform: translateY(-75px);

}

}

.box {

width: 100px;

height: 100px;

background-color: blue;

transition: background-color 1s;

animation: bounce 2s infinite;

}

.box:hover {

background-color: red;

}

</style>

</head>

<body>

<div class="box"></div>

</body>

</html>

In this example:

* The bounce animation makes the element bounce up and down.
* The .box class applies the bounce animation and a transition on the background-color.
* When you hover over the .box, the background color changes to red over 1 second, while the bounce animation continues to run.

**Examples**

**1. Button Hover Effect with Transition and Animation**

This example combines a transition on hover with an animation to create a ripple effect on a button.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Button Hover Effect</title>

<style>

.button {

position: relative;

padding: 15px 30px;

background-color: #007bff;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

overflow: hidden;

transition: background-color 0.3s ease;

}

.button:hover {

background-color: #0056b3;

}

.button::after {

content: "";

position: absolute;

width: 100%;

height: 100%;

top: 0;

left: 0;

background: rgba(255, 255, 255, 0.5);

border-radius: 50%;

opacity: 0;

transform: scale(0);

transition: transform 0.5s, opacity 0.5s;

}

.button:hover::after {

opacity: 1;

transform: scale(4);

}

</style>

</head>

<body>

<button class="button">Hover Me</button>

</body>

</html>

**2. Image Zoom Effect with Transition and Animation**

This example creates a zoom-in effect on an image when it is hovered over, combined with an animation for a smooth transition.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Image Zoom Effect</title>

<style>

.image-container {

width: 300px;

overflow: hidden;

}

.image-container img {

width: 100%;

transition: transform 0.5s ease;

}

.image-container img:hover {

transform: scale(1.1);

}

.image-container img {

animation: fadeIn 1s ease-in-out;

}

@keyframes fadeIn {

from {

opacity: 0;

}

to {

opacity: 1;

}

}

</style>

</head>

<body>

<div class="image-container">

<img src="https://via.placeholder.com/300" alt="Sample Image">

</div>

</body>

</html>

**3. Card Flip Effect with Transition and Animation**

This example creates a card flip effect where the card rotates on hover, combining transitions and animations for a smooth experience.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Card Flip Effect</title>

<style>

.card {

width: 200px;

height: 300px;

perspective: 1000px;

}

.card-inner {

width: 100%;

height: 100%;

transition: transform 0.6s;

transform-style: preserve-3d;

position: relative;

}

.card:hover .card-inner {

transform: rotateY(180deg);

}

.card-front, .card-back {

width: 100%;

height: 100%;

position: absolute;

backface-visibility: hidden;

border-radius: 10px;

}

.card-front {

background-color: #007bff;

color: white;

display: flex;

align-items: center;

justify-content: center;

font-size: 24px;

}

.card-back {

background-color: #f2f2f2;

color: #333;

transform: rotateY(180deg);

display: flex;

align-items: center;

justify-content: center;

font-size: 24px;

}

</style>

</head>

<body>

<div class="card">

<div class="card-inner">

<div class="card-front">

Front

</div>

<div class="card-back">

Back

</div>

</div>

</div>

</body>

</html>

**4. Navigation Menu with Transition and Animation**

This example creates a navigation menu that slides in from the left when a button is clicked, combining transitions and animations for smooth opening and closing.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Navigation Menu</title>

<style>

body {

margin: 0;

font-family: Arial, sans-serif;

}

.menu-button {

position: absolute;

top: 20px;

left: 20px;

background-color: #007bff;

color: white;

border: none;

padding: 10px 20px;

cursor: pointer;

}

.menu {

position: fixed;

top: 0;

left: -250px;

width: 250px;

height: 100%;

background-color: #333;

color: white;

padding: 20px;

transition: left 0.3s ease;

}

.menu.open {

left: 0;

}

.menu-item {

margin: 15px 0;

opacity: 0;

animation: fadeInMenu 0.5s forwards;

}

@keyframes fadeInMenu {

from {

opacity: 0;

transform: translateX(-20px);

}

to {

opacity: 1;

transform: translateX(0);

}

}

</style>

</head>

<body>

<button class="menu-button" onclick="toggleMenu()">Menu</button>

<div class="menu" id="menu">

<div class="menu-item">Home</div>

<div class="menu-item">About</div>

<div class="menu-item">Services</div>

<div class="menu-item">Contact</div>

</div>

<script>

function toggleMenu() {

const menu = document.getElementById('menu');

menu.classList.toggle('open');

}

</script>

</body>

</html>

**5. Loading Spinner with Transition and Animation**

This example creates a loading spinner that grows and shrinks, combining transitions and animations for a smooth effect.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Loading Spinner</title>

<style>

.spinner {

width: 50px;

height: 50px;

border: 5px solid rgba(0, 0, 0, 0.1);

border-top: 5px solid #007bff;

border-radius: 50%;

animation: spin 1s linear infinite, pulse 1.5s infinite;

}

@keyframes spin {

from {

transform: rotate(0deg);

}

to {

transform: rotate(360deg);

}

}

@keyframes pulse {

0%, 100% {

transform: scale(1);

}

50% {

transform: scale(1.2);

}

}

</style>

</head>

<body>

<div class="spinner"></div>

</body>

</html>

**Transforms and Transitions**: Explore CSS transforms to manipulate the shape, size, and position of elements, and transitions to create smooth effects when properties change.