Kodutöö 1 - ChatGPT

Karmen Klaasen
IFIFB/23.DT

Please improve my code so that the website that shows the current time and date would look visually appealing and have interactive elements for the user

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <script src="clock.js" defer></script>
   <link rel="stylesheet" href="style.css">
   <title>Kell</title>
</head>
<body>
   <div id="clockContainer">
      <div id="clock">
         <div id="hours">tunnid</div>
         <div id="minutes">minutid</div>
         <div id="seconds">sekundid</div>
      </div>
      <div id="date">
         <div id="day">kuupäev</div>
         <div id="month">kuu</div>
         <div id="year">aasta</div>
      </div>
      <div id="today">tänane päev</div>
   </div>
</body>
</html>
```

```
let hours, minutes, seconds, date, day, month, year;
let months = ["Jaanuar", "Veebruar", "Märts", "Aprill", "Mai", "Juuni", "Juuli", "August",
"September", ];
let days = ["Pühapäev", "Esmaspäev", "Teisipäev", "Kolmapäev", "Neljapäev", "Reede",
"Laupäev"]
window.addEventListener('keypress', textSize);
let timeFontSize = 4;
```

```
document.getElementById('clock').style.fontSize = timeFontSize + "em";


function textSize(e){
    console.log(e.keyCode);
    if(e.keyCode == 45){
        timeFontSize = timeFontSize - 0.1;
        document.getElementById('clock').style.fontSize = timeFontSize + "em";
    }
    if(e.keyCode == 43){
        timeFontSize = timeFontSize + 0.1;
        document.getElementById('clock').style.fontSize = timeFontSize + "em";
    }
}

function updateClock(){
    date = new Date();
    console.log(date);
    hours = date.getHours();
    minutes = date.getMinutes();
    seconds = date.getSeconds();
    console.log(hours, minutes, seconds);
    if(hours < 10) {
        hours = "0" + hours;
    }
    if(minutes < 10) {
        minutes = "0" + minutes;
    }
    if(seconds < 10) {
        seconds = "0" + seconds
    }
    document.getElementById('hours').innerHTML = hours;
    document.getElementById('minutes').innerHTML = ":" + minutes;
    document.getElementById('seconds').innerHTML = ":" + seconds;
}

function updateDate(){
    date = new Date();

    day = date.getDate();
    month = date.getMonth();
    year = date.getFullYear();
```

```
    dayValue = date.getDay();

    document.getElementById('day').innerHTML = day;
    document.getElementById('month').innerHTML = months[month];
    document.getElementById('year').innerHTML = year;

    document.getElementById('today').innerHTML = days[dayValue];
}

updateClock();
updateDate();
setInterval(updateClock, 1000);

body{
    background-color: cadetblue;
}

#clockContainer{
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    height: 80dvh;
}

#clock, #date{
    display: flex;
}

#date>div{
    padding-right: 10px;
}

#hours{
    background-color: aqua;
}

#minutes{
    background-color: bisque;
}

#seconds{
```

```
    background-color: gainsboro;
}
```
To improve the visual appeal and interactivity of your website, you can consider the following enhancements:

1. **Add Transitions**:
   - Apply smooth transitions for the clock and date elements to make the changes more visually appealing.

2. **Style Enhancements**:
   - Improve the layout, font, and colors to make the website more visually appealing.
   - Use a modern font and adjust the sizes and colors of the elements for better readability.

3. **Interactive Elements**:
   - Implement interactive features such as buttons to control the font size or color scheme dynamically.
   - Add hover effects or animations to make the website more engaging.

4. **Responsive Design**:
   - Ensure that your website looks good and functions well on various screen sizes and devices.

Here's an improved version of your HTML and CSS with some enhancements:

HTML:
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="style.css">
    <title>Kell</title>
</head>
<body>
    <div id="clockContainer">
        <div id="clock">
            <div id="time"></div>
        </div>
        <div id="date">
            <div id="day"></div>
```

```
        <div id="month"></div>
        <div id="year"></div>
        <div id="today"></div>
      </div>
    </div>
    <script src="clock.js" defer></script>
</body>
</html>
```

JavaScript (clock.js):
```javascript
let timeElement = document.getElementById('time');
let dateElement = document.getElementById('date');
let dayElement = document.getElementById('day');
let monthElement = document.getElementById('month');
let yearElement = document.getElementById('year');
let todayElement = document.getElementById('today');

let months = ["Jaanuar", "Veebruar", "Märts", "Aprill", "Mai", "Juuni", "Juuli", "August",
"September"];
let days = ["Pühapäev", "Esmaspäev", "Teisipäev", "Kolmapäev", "Neljapäev", "Reede",
"Laupäev"];

function updateClock() {
    let date = new Date();
    let hours = date.getHours().toString().padStart(2, '0');
    let minutes = date.getMinutes().toString().padStart(2, '0');
    let seconds = date.getSeconds().toString().padStart(2, '0');

    timeElement.textContent = `${hours}:${minutes}:${seconds}`;
}

function updateDate() {
    let date = new Date();
    let day = date.getDate();
    let month = date.getMonth();
    let year = date.getFullYear();
    let dayValue = date.getDay();

    dayElement.textContent = day;
    monthElement.textContent = months[month];
```

```
    yearElement.textContent = year;
    todayElement.textContent = days[dayValue];
}

updateClock();
updateDate();
setInterval(updateClock, 1000);
```

CSS (style.css):
```css
body {
    background-color: #f0f0f0;
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
}

#clockContainer {
    text-align: center;
}

#time {
    font-size: 3rem;
    color: #333;
    margin-bottom: 20px;
    transition: font-size 0.3s ease;
}

#date {
    font-size: 1.2rem;
    color: #666;
}

#date > div {
    display: inline-block;
    margin-right: 10px;
}
```

```
#today {
  font-weight: bold;
}

/* Add hover effects or animations as per your preference */
```

This enhanced version adds smoother transitions, improves the layout, and adjusts the font sizes and colors for better readability and visual appeal. Additionally, it maintains the functionality of the clock and date display while making the code more organized and easier to maintain. You can further enhance it with additional CSS effects or interactive elements based on your requirements and preferences.

How can I insert a box around the time and date and then separate the time and date so that initially the site shows the time box and when the user clicks on the box it slides into the date box?

To implement the functionality where initially only the time is displayed, and upon clicking, it slides into the date box, you need to add some JavaScript for handling the click event and CSS for creating the slide effect.

Here's the updated code:

HTML:
```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="style.css">
  <title>Kell</title>
</head>
<body>
  <div id="clockContainer">
    <div id="timeBox" class="box" onclick="slideToDate()">
      <div id="time"></div>
    </div>
    <div id="dateBox" class="box hidden">
      <div id="day"></div>
      <div id="month"></div>
```

```
        <div id="year"></div>
        <div id="today"></div>
      </div>
    </div>
    <script src="clock.js" defer></script>
</body>
</html>
```

JavaScript (clock.js):
```javascript
let timeElement = document.getElementById('time');
let dateElement = document.getElementById('date');
let dayElement = document.getElementById('day');
let monthElement = document.getElementById('month');
let yearElement = document.getElementById('year');
let todayElement = document.getElementById('today');
let timeBox = document.getElementById('timeBox');
let dateBox = document.getElementById('dateBox');

let months = ["Jaanuar", "Veebruar", "Märts", "Aprill", "Mai", "Juuni", "Juuli", "August",
"September"];
let days = ["Pühapäev", "Esmaspäev", "Teisipäev", "Kolmapäev", "Neljapäev", "Reede",
"Laupäev"];

function updateClock() {
    let date = new Date();
    let hours = date.getHours().toString().padStart(2, '0');
    let minutes = date.getMinutes().toString().padStart(2, '0');
    let seconds = date.getSeconds().toString().padStart(2, '0');

    timeElement.textContent = `${hours}:${minutes}:${seconds}`;
}

function updateDate() {
    let date = new Date();
    let day = date.getDate();
    let month = date.getMonth();
    let year = date.getFullYear();
    let dayValue = date.getDay();

    dayElement.textContent = day;
```

```
    monthElement.textContent = months[month];
    yearElement.textContent = year;
    todayElement.textContent = days[dayValue];
}

function slideToTime() {
    timeBox.classList.remove('hidden');
    dateBox.classList.add('hidden');
}

function slideToDate() {
    timeBox.classList.add('hidden');
    dateBox.classList.remove('hidden');
}

updateClock();
updateDate();
setInterval(updateClock, 1000);
```

CSS (style.css):
```css
body {
    background-color: #f0f0f0;
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
}

#clockContainer {
    text-align: center;
}

.box {
    border: 2px solid #333;
    padding: 10px;
    cursor: pointer;
    transition: width 0.5s, height 0.5s, padding 0.5s;
```

```
    margin-bottom: 20px;
}

.hidden {
    display: none;
}

#time {
    font-size: 3rem;
    color: #333;
}

#date {
    font-size: 1.2rem;
    color: #666;
}

#date > div {
    display: inline-block;
    margin-right: 10px;
}

#today {
    font-weight: bold;
}
```

In this updated code:

- The time and date elements are placed inside separate box containers (`timeBox` and `dateBox`).
- Initially, the time box is visible, and the date box is hidden using the `hidden` class.
- Click event handlers (`slideToTime()` and `slideToDate()`) are added to toggle the visibility of the time and date boxes.
- CSS transitions are applied to create a smooth sliding effect when switching between the time and date boxes.

Now, when the user clicks on the time box, it will slide out, and the date box will slide in, revealing the date information.

Please add different interactive options for the user, so in total the clock and time website would have 6 interactive actions.

To enhance the interactivity of your clock and date website, let's add a few more interactive actions. Here's how we can achieve this:

1. **Change Clock Format**: Allow users to toggle between 12-hour and 24-hour clock formats.
2. **Toggle Seconds**: Allow users to toggle the visibility of seconds in the clock.
3. **Change Background Color**: Allow users to change the background color of the webpage.

Here's the modified JavaScript code implementing these functionalities:

```javascript
let timeElement = document.getElementById('time');
let dayElement = document.getElementById('day');
let monthElement = document.getElementById('month');
let yearElement = document.getElementById('year');
let todayElement = document.getElementById('today');
let timeBox = document.getElementById('timeBox');
let dateBox = document.getElementById('dateBox');

let months = ["Jaanuar", "Veebruar", "Märts", "Aprill", "Mai", "Juuni", "Juuli", "August", "September"];
let days = ["Pühapäev", "Esmaspäev", "Teisipäev", "Kolmapäev", "Neljapäev", "Reede", "Laupäev"];

let showingTime = true;
let fontSize = 3; // Initial font size
let clockFormat = 24; // Default 24-hour clock
let showSeconds = true; // Show seconds by default
let backgroundColors = ["#f0f0f0", "#d3d3d3", "#b0c4de", "#add8e6"];
let currentBackgroundColor = 0;

function updateClock() {
    let date = new Date();
    let hours = date.getHours();
    let minutes = date.getMinutes().toString().padStart(2, '0');
    let seconds = date.getSeconds().toString().padStart(2, '0');

    if (clockFormat === 12) {
        hours = (hours % 12) || 12;
    }
```

```
    let timeString = `${hours}:${minutes}`;
    if (showSeconds) {
        timeString += `:${seconds}`;
    }

    timeElement.textContent = timeString;
    timeElement.style.fontSize = fontSize + 'rem';
}

function updateDate() {
    let date = new Date();
    let day = date.getDate();
    let month = date.getMonth();
    let year = date.getFullYear();
    let dayValue = date.getDay();

    dayElement.textContent = day;
    monthElement.textContent = months[month];
    yearElement.textContent = year;
    todayElement.textContent = days[dayValue];

    // Apply font size to date elements
    dayElement.style.fontSize = fontSize * 0.4 + 'rem';
    monthElement.style.fontSize = fontSize * 0.4 + 'rem';
    yearElement.style.fontSize = fontSize * 0.4 + 'rem';
    todayElement.style.fontSize = fontSize * 0.4 + 'rem';
}

function toggleBoxes() {
    if (showingTime) {
        timeBox.classList.add('hidden');
        dateBox.classList.remove('hidden');
    } else {
        timeBox.classList.remove('hidden');
        dateBox.classList.add('hidden');
    }
    showingTime = !showingTime;
}

function changeFontSize(e) {
    if (e.key === 'ArrowUp') {
        fontSize += 0.1;
```

```
    } else if (e.key === 'ArrowDown' && fontSize > 3) {
        fontSize -= 0.1;
    }
    updateClock();
    updateDate();
}

function toggleClockFormat() {
    clockFormat = (clockFormat === 24) ? 12 : 24;
    updateClock();
}

function toggleShowSeconds() {
    showSeconds = !showSeconds;
    updateClock();
}

function changeBackgroundColor() {
    currentBackgroundColor = (currentBackgroundColor + 1) %
backgroundColors.length;
    document.body.style.backgroundColor =
backgroundColors[currentBackgroundColor];
}

updateClock();
updateDate();
setInterval(updateClock, 1000);

timeBox.addEventListener('click', toggleBoxes);
dateBox.addEventListener('click', toggleBoxes);
document.addEventListener('keydown', changeFontSize);
```

In this code:

- We've added functions to toggle the clock format between 12-hour and 24-hour (`toggleClockFormat`), toggle the display of seconds (`toggleShowSeconds`), and change the background color of the webpage (`changeBackgroundColor`).
- Each function listens for different events or user interactions to trigger the corresponding action.

- The clock format, show seconds, and background color are controlled by variables that can be toggled or changed based on user input.

You can now integrate the respective event listeners and connect these functions with HTML elements to enable user interaction and enhance the functionality of your clock and date website.

What else can I use instead of 'margin-top: auto;' to keep the clockcontainer in the middle of the page and the buttons in the bottom of the page?

Instead of using `margin-top: auto;` to push the buttons container to the bottom of the page, you can achieve the same effect by adjusting the `margin-top` of the clock container. Here's how you can modify your CSS:

```css
body {
    background-color: #f0f0f0;
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    height: 100vh;
}

#clockContainer {
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    margin: auto 0 20px; /* Center clock container vertically and add space at the bottom */
}

.box {
    border: 2px solid #333;
    padding: 10px;
    cursor: pointer;
    transition: width 0.5s, height 0.5s, padding 0.5s;
    margin-bottom: 20px;
```

```
}

.hidden {
    display: none;
}

#time {
    font-size: 3rem;
    color: #333;
}

#date {
    font-size: 1.2rem;
    color: #666;
}

#date > div {
    display: inline-block;
    margin-right: 10px;
}

#today {
    font-weight: bold;
}

#buttonContainer {
    display: flex;
    justify-content: center;
    align-items: center;
    margin-bottom: 20px; /* Adds some space between the clock and buttons */
}
```

In this modification, `margin: auto 0 20px;` is used for the clock container to center it vertically within the `body` and also add space at the bottom. Then, the `#buttonContainer` is positioned at the bottom using `margin-bottom: 20px;` without relying on `margin-top: auto;`.

Please add a gradient to the background that would include colors like pink, purple, blue

To add a gradient background that includes colors like pink, purple, and blue, you can update the body's background style as follows:

```css
body {
    background: linear-gradient(to bottom right, #ff85a2, #a683ee, #6f7cff); /*
Gradient background */
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    height: 100vh;
}

/* Rest of your CSS styles remain unchanged */
```

In this update:

- The `background` property is used to set a linear gradient background.
- The `linear-gradient` function is used to define the gradient.
- The gradient starts from the top left corner (`to bottom right`) and includes colors `#ff85a2` (pink), `#a683ee` (purple), and `#6f7cff` (blue).
- This creates a smooth transition of colors from pink to purple to blue diagonally across the background.

Please improve the code so it would change the background color into a different gradient color

To improve the code and change the background color into a different gradient color, you can modify the CSS section where the background gradient is defined. Here's the updated code:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <link rel="stylesheet" href="style.css">
    <title>Kell</title>
</head>
<body>
    <div id="clockContainer">
        <div id="timeBox" class="box" onclick="slideToDate()">
            <div id="time"></div>
        </div>
        <div id="dateBox" class="box hidden">
            <div id="day"></div>
            <div id="month"></div>
            <div id="year"></div>
            <div id="today"></div>
        </div>
    </div>

    <div id="buttonContainer">
        <button onclick="toggleClockFormat()">Toggle Clock Format</button>
        <button onclick="toggleShowSeconds()">Toggle Seconds</button>
        <button onclick="changeBackgroundColor()">Change Background
Color</button>
        <button onclick="toggleLanguage()">Toggle Language</button>
    </div>
    <script src="clock.js" defer></script>
</body>
</html>

<script>
    let timeElement = document.getElementById('time');
    let dayElement = document.getElementById('day');
    let monthElement = document.getElementById('month');
    let yearElement = document.getElementById('year');
    let todayElement = document.getElementById('today');
    let timeBox = document.getElementById('timeBox');
    let dateBox = document.getElementById('dateBox');

    let months = ["January", "February", "March", "April", "May", "June", "July", "August",
"September"];
    let days = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday",
"Saturday"];
    let monthsEstonian = ["Jaanuar", "Veebruar", "Märts", "Aprill", "Mai", "Juuni", "Juuli",
"August", "September"];
```

```
    let daysEstonian = ["Pühapäev", "Esmaspäev", "Teisipäev", "Kolmapäev",
"Neljapäev", "Reede", "Laupäev"];


    let showingTime = true;
    let fontSize = 3; // Initial font size
    let clockFormat = 24; // Default 24-hour clock
    let showSeconds = true; // Show seconds by default
    let backgroundColors = ["linear-gradient(to bottom right, #ff85a2, #a683ee,
#6f7cff)", "linear-gradient(to bottom right, #85ff8a, #83eea6, #7cff6f)",
"linear-gradient(to bottom right, #85a2ff, #83ee83, #7c6fff)", "linear-gradient(to
bottom right, #ffe985, #ee83a6, #ff6f7c)"];
    let currentBackgroundColor = 0;
    let isEnglish = false; // Default language is Estonian

    function updateClock() {
        let date = new Date();
        let hours = date.getHours();
        let minutes = date.getMinutes().toString().padStart(2, '0');
        let seconds = date.getSeconds().toString().padStart(2, '0');

        if (clockFormat === 12) {
            period = hours >= 12 ? 'PM' : 'AM';
            hours = (hours % 12) || 12;
        }

        let timeString = `${hours}:${minutes}`;
        if (showSeconds) {
            timeString += `:${seconds}`;
        }

        if (clockFormat === 12) {
            timeString += ` ${period}`;
        }

        timeElement.textContent = timeString;
        timeElement.style.fontSize = fontSize + 'rem';
    }

    function updateDate() {
        let date = new Date();
        let day = date.getDate();
```

```
        let month = date.getMonth();
        let year = date.getFullYear();
        let dayValue = date.getDay();

        dayElement.textContent = day;
        monthElement.textContent = isEnglish ? months[month] :
monthsEstonian[month];
        yearElement.textContent = year;
        todayElement.textContent = isEnglish ? days[dayValue] :
daysEstonian[dayValue];

        // Apply font size to date elements
        dayElement.style.fontSize = fontSize * 0.4 + 'rem';
        monthElement.style.fontSize = fontSize * 0.4 + 'rem';
        yearElement.style.fontSize = fontSize * 0.4 + 'rem';
        todayElement.style.fontSize = fontSize * 0.4 + 'rem';
    }

    function toggleBoxes() {
        if (showingTime) {
            timeBox.classList.add('hidden');
            dateBox.classList.remove('hidden');
        } else {
            timeBox.classList.remove('hidden');
            dateBox.classList.add('hidden');
        }
        showingTime = !showingTime;
    }

    function changeFontSize(e) {
        if (e.key === 'ArrowUp') {
            fontSize += 0.1;
        } else if (e.key === '
```

Please use a halo effect for the time and date box

To create a halo effect for the time and date boxes, you can add a box-shadow CSS property to the `.box` class. Here's how you can modify the CSS:

```css
.box {
    border: 2px solid #333;
```

```
    padding: 10px;
    cursor: pointer;
    transition: width 0.5s, height 0.5s, padding 0.5s;
    margin-bottom: 20px;
    box-shadow: 0 0 20px rgba(255, 255, 255, 0.7); /* Halo effect */
}
```

This CSS code adds a white halo effect around the time and date boxes. Adjust the values (such as the color and spread radius) to achieve the desired effect.

Make sure to include this CSS code in your `style.css` file or within a `<style>` tag in the `<head>` section of your HTML document.

How can I add an upper limit so the size can't be changed bigger forever?

To add an upper limit to the font size so that it cannot be increased indefinitely, you can define a maximum font size and check if the current font size exceeds that limit before increasing it further. Here's how you can modify the `changeFontSize` function to include an upper limit:

```javascript
// Define the maximum font size
const maxFontSize = 5;

function changeFontSize(e) {
    if (e.key === 'ArrowUp' && fontSize < maxFontSize) {
        fontSize += 0.1;
    } else if (e.key === 'ArrowDown' && fontSize > 3) {
        fontSize -= 0.1;
    }
    updateClock();
    updateDate();
}
```

In this modification:

- We introduced a constant `maxFontSize` with a value of `5`, representing the upper limit of the font size.

- In the `changeFontSize` function, we added a condition to the `ArrowUp` key event that checks if the current font size is less than `maxFontSize` before increasing it further.
- If the current font size is already at the maximum (`maxFontSize`), it won't increase anymore, preventing it from going beyond the specified upper limit.

Please improve the code so that in addition to showing the time and date separately when clicking on the box, it would also have a third option of showing the time and date together

To add a third option to show the time and date together when clicking on the box, you can modify the `toggleBoxes` function to toggle between displaying the time, date, and both time and date. Here's how you can do it:

```javascript
function toggleBoxes() {
  if (showingTime && !dateBox.classList.contains('hidden')) {
    timeBox.classList.add('hidden');
    dateBox.classList.remove('hidden');
  } else if (!showingTime && dateBox.classList.contains('hidden')) {
    timeBox.classList.remove('hidden');
    dateBox.classList.add('hidden');
  } else {
    timeBox.classList.remove('hidden');
    dateBox.classList.remove('hidden');
  }
  showingTime = !showingTime;
}
```

In this modification:

- We added an additional condition to check if both the time and date are being displayed. If so, when clicking on the box, it will show both the time and date together.
- We removed the `hidden` class from both the time and date boxes if they were hidden, allowing both elements to be visible simultaneously when the third option is chosen.

With these changes, when you click on the box, it will cycle between showing the time, the date, and both the time and date together.

How can I add a switch for a dark version that would change the background into darker gradients?

To add a switch for a dark version that changes the background into darker gradients, you can implement a simple toggle switch in your HTML and then use JavaScript to change the background gradients accordingly. Here's how you can do it:

1. **HTML for Toggle Switch**:
   Add a toggle switch to your HTML markup.

```html
<input type="checkbox" id="darkModeToggle">
<label for="darkModeToggle" id="darkModeLabel">Dark Mode</label>
```

2. **JavaScript to Toggle Dark Mode**:
   Write JavaScript code to toggle the dark mode when the switch is clicked. This code will change the background gradients based on whether the dark mode is enabled or disabled.

```javascript
// JavaScript code
let darkModeToggle = document.getElementById('darkModeToggle');
let darkModeLabel = document.getElementById('darkModeLabel');

darkModeToggle.addEventListener('change', function() {
    if (this.checked) {
        enableDarkMode();
    } else {
        disableDarkMode();
    }
});

function enableDarkMode() {
    document.body.classList.add('dark-mode');
    darkModeLabel.textContent = 'Light Mode';
}

function disableDarkMode() {
    document.body.classList.remove('dark-mode');
    darkModeLabel.textContent = 'Dark Mode';
```

}
```

3. **CSS for Dark Mode Styling**:
   Add CSS rules to define the background gradients for the dark mode.

```css
/* CSS code */
.dark-mode {
    background: linear-gradient(to bottom right, #333333, #000000); /* Dark gradient background */
    color: #ffffff; /* Light text color */
}
```

With this setup, when the toggle switch is clicked, the background gradients will change to darker tones, and the text color will change to light to maintain readability. Conversely, clicking the switch again will revert the page to its original light mode.

You can adjust the gradient colors and other styles according to your design preferences.
```