Flex Panel Gallery

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <title>Flex Panels 💪</title>
  <link href='https://fonts.googleapis.com/css?family=Amatic+SC'
rel='stylesheet' type='text/css'>
  <link rel="icon" href="https://fav.farm/☑" />
</head>

<body>
  <style>
    html {
      box-sizing: border-box;
      background: #ffc600;
      font-family: 'helvetica neue';
      font-size: 20px;
      font-weight: 200;
    }

    body {
      margin: 0;
    }

    *,
    *:before,
    *:after {
      box-sizing: inherit;
    }

    .panels {
      min-height: 100vh;
      overflow: hidden;
      display: flex;
    }

    .panel {
      background: #6B0F9C;
      box-shadow: inset 0 0 0 5px rgba(255, 255, 255, 0.1);
      color: white;
      text-align: center;
      align-items: center;
      /* Safari transitionend event.propertyName === flex */
      /* Chrome + FF transitionend event.propertyName === flex-grow */
```

```
      transition: font-size 0.7s cubic-bezier(0.61, -0.19, 0.7, -0.11),
        flex 0.7s cubic-bezier(0.61, -0.19, 0.7, -0.11),
        background-color 0.2s cubic-bezier(0.61, -0.19, 0.7, -0.11);
      font-size: 20px;
      background-size: cover;
      background-position: center;
      flex: 1;
      justify-content: center;
      display: flex;
      flex-direction: column;
    }

    .panel1 {
      background-image: url(https://source.unsplash.com/gYl-
UtwNg_I/1500x1500);
    }

    .panel2 {
      background-image:
url(https://source.unsplash.com/rFKUFzjPYiQ/1500x1500);
    }

    .panel3 {
      background-image: url(https://images.unsplash.com/photo-1465188162913-
8fb5709d6d57?ixlib=rb-
0.3.5&q=80&fm=jpg&crop=faces&cs=tinysrgb&w=1500&h=1500&fit=crop&s=967e8a713a4e
395260793fc8c802901d);
    }

    .panel4 {
      background-image:
url(https://source.unsplash.com/ITjiVXcwVng/1500x1500);
    }

    .panel5 {
      background-image:
url(https://source.unsplash.com/3MNzGlQM7qs/1500x1500);
    }

    /* Flex Items */
    .panel>* {
      margin: 0;
      width: 100%;
      transition: transform 0.5s;
      flex: 1 0 auto;
      display: flex;
      justify-content: center;
      align-items: center;
    }
```

```
  .panel>*:first-child {
    transform: translateY(-100%);
  }

  .panel.open-active>*:first-child {
    transform: translateY(0);
  }

  .panel>*:last-child {
    transform: translateY(100%);
  }

  .panel.open-active>*:last-child {
    transform: translateY(0);
  }

  .panel p {
    text-transform: uppercase;
    font-family: 'Amatic SC', cursive;
    text-shadow: 0 0 4px rgba(0, 0, 0, 0.72), 0 0 14px rgba(0, 0, 0, 0.45);
    font-size: 2em;
  }

  .panel p:nth-child(2) {
    font-size: 4em;
  }

  .panel.open {
    flex: 5;
    font-size: 40px;
  }

  @media only screen and (max-width: 600px) {
    .panel p {
      font-size: 1em;
    }
  }
</style>


<div class="panels">
  <div class="panel panel1">
    <p>Hey</p>
    <p>Let's</p>
    <p>Dance</p>
  </div>
  <div class="panel panel2">
    <p>Give</p>
```

```
      <p>Take</p>
      <p>Receive</p>
    </div>
    <div class="panel panel3">
      <p>Experience</p>
      <p>It</p>
      <p>Today</p>
    </div>
    <div class="panel panel4">
      <p>Give</p>
      <p>All</p>
      <p>You can</p>
    </div>
    <div class="panel panel5">
      <p>Life</p>
      <p>In</p>
      <p>Motion</p>
    </div>
  </div>

  <script>
    const panels = document.querySelectorAll('.panel');

    function toggleOpen() {
      console.log("Hello");
      this.classList.toggle('open');
    }

    function toggleActive(e) {
      console.log(e.propertyName);
      if (e.propertyName.includes('flex')) {

        /* We didn't use "e.propertyname === 'flex-grow',
       but because some browsers interpretend it as 'flex' and others as
'flex-grow',
        we had to use a different method" */

        this.classList.toggle('open-active');
      }
    }

    panels.forEach(panel => panel.addEventListener('click', toggleOpen));
    panels.forEach(panel => panel.addEventListener('transitionend',
toggleActive));
  </script>

</body>

</html>
```

Below is a brief explanation of the exercise that I have made for this challenge:

This exercise serves as practice for working with CSS and JavaScript in web development. By utilizing CSS's flexbox for responsive design and transitions and incorporating JavaScript for interactive features like toggling classes on panel clicks.

The project is a practical demonstration of combining CSS and JavaScript to create a visually appealing and interactive web interface with flex panels.

CSS

1. **Global Styles:**
   The CSS begins with styling for the entire HTML document. It sets the box model to border-box, defines a background color, and specifies font styles.

2. **Panel Styles:**
   Styles for the panel class define the appearance of each panel. It includes background colors, box shadow, text color, and a transition effect for changes in font size, flex, and background color. Each panel is configured as a flex container with a vertical flex direction.

3. **Individual Panel Backgrounds:**
   Each panel (panel1 to panel5) is given a unique background image using the background-image property.

4. **Flex Item Styles:**
   Styles for the child elements of each panel (the <p> elements) are defined. They are set to take the full width of the panel, have a transition effect for transformations, and are initially transformed to be outside the panel's visible area.

5. **Media Query:**
   A media query is used to adjust the font size of the panel text for screens with a maximum width of 600 pixels.

JavaScript

6. **Panel Selection:**
   The JavaScript code selects all elements with the class "panel" and stores them in the panels variable.

7. **Toggle Open Function:**
   The toggleOpen function is defined to toggle the "open" class on a panel when it is clicked. This class triggers a change in the panel's flex properties and initiates the transition effect.

8. **Toggle Active Function:**
   The toggleActive function is defined to toggle the "open-active" class when a transition on the flex property ends. This class is responsible for the final visual adjustments after the panel is opened.

9. **Event Listeners:**
   Event listeners are added to each panel. The toggleOpen function is called when a panel is clicked, and the toggleActive function is called when a transition on the flex property completes.

What I have learned

From this challenge, I have learned how to use CSS Flexbox for flexible layouts and responsive design. In JavaScript, I gained insights into handling user interactions, using event listeners to toggle classes for smooth animations and interactivity.