1. CSS Tooltips

Explanation

CSS tooltips show extra information when the user hovers over an element. They are used to explain icons, buttons, or text without cluttering the page.

Example

```
<span class="tip">Hover me
  <span class="tooltip-text">This is a tooltip</span>
</span>
.tip {
  position: relative;
}

.tooltip-text {
  visibility: hidden;
  background-color: black;
  color: white;
  padding: 5px;
  position: absolute;
}

.tip:hover .tooltip-text {
  visibility: visible;
}
```

Use: Help messages, hints, icon descriptions.

---

2. CSS Image Styling

Explanation

CSS image styling is used to change the appearance of images like size, border, shape, and shadow.

Example

```
img {
  width: 200px;
  border: 2px solid black;
  border-radius: 10px;
}
```

Use: Profile pictures, product images, galleries.

---

3. CSS Image Modal

Explanation

An image modal shows a large image in a popup when the user clicks on a small image.

Example

```
img {
  cursor: pointer;
}
```

```
<img src="small.jpg">
```

Use: Image previews, galleries, zoom effects.

---

4. CSS Image Centering

Explanation

CSS image centering places an image in the center of a page or container.

Example

```
img {
  display: block;
  margin: auto;
}
```

Use: Banners, logos, profile images.

---

5. CSS Image Filters

Explanation

CSS image filters apply visual effects like blur, brightness, or grayscale to images.

Example

```
img {
  filter: grayscale(100%);
}
```

Use: Hover effects, photo effects, design styling.

---

6. CSS Image Shapes

Explanation

CSS image shapes change the shape of images using properties like border-radius or clip-path.

Example

```
img {
  border-radius: 50%;
}
```

Use: Circular profile pictures, creative layouts.

CSS object-fit Property

The CSS object-fit property is used to specify how an <img> or <video> should be resized to fit its container.

This property can take one of the following values:

- fill - This is default. Does not preserve the aspect ratio. The image is resized to fill the container (the image will be stretched or squeezed to fit).
- cover - Preserves the aspect ratio, and the image fills the container. Cuts overflowing content if needed.
- contain - Preserves the aspect ratio, and fits the image inside the container, without cutting - leaves empty space if needed.
- none - The image is not resized.
- scale-down - the image is scaled down to the smallest version of none or contain.

Using object-fit: fill;

The object-fit: fill; value does not preserve the aspect ratio, and the image is resized to fill the container (the image will be stretched or squeezed to fit):

Example

```
.image-container {
  width: 200px;
  height: 300px;
  border: 1px solid black;
```

```
  margin-bottom: 25px;
}

.image-container img {
  width: 100%;
  height: 100%;
  object-fit: fill;
}
```

Using object-fit: cover;

The object-fit: cover; value preserves the aspect ratio, and the image fills the container. The image will be clipped to fit:

Example

```
.image-container {
  width: 200px;
  height: 300px;
  border: 1px solid black;
  margin-bottom: 25px;
}

.image-container img {
  width: 100%;
  height: 100%;
  object-fit: cover;
}
```

Using object-fit: contain;

The object-fit: contain; value preserves the aspect ratio, and fits the image inside the container, without cutting - will leave empty space if needed:

Example

```
.image-container {
  width: 200px;
  height: 300px;
  border: 1px solid black;
  margin-bottom: 25px;
}

.image-container img {
  width: 100%;
  height: 100%;
  object-fit: contain;
}
```

Using object-fit: none;

The object-fit: none; value does not resize or scale the image:

Example

```
.image-container {
  width: 200px;
  height: 300px;
  border: 1px solid black;
  margin-bottom: 25px;
}

.image-container img {
  width: 100%;
  height: 100%;
  object-fit: none;
}
```

Using object-fit: scale-down;

The object-fit: scale-down; value scales the image down to the smallest version of none or contain:

Example

```
.image-container {
  width: 200px;
  height: 300px;
  border: 1px solid black;
  margin-bottom: 25px;
}

.image-container img {
  width: 100%;
  height: 100%;
  object-fit: scale-down;
}
```

CSS object-position Property

The CSS object-position property is used together with the object-fit property to specify how an <img> or <video> should be positioned with x/y coordinates within its container.

The first value controls the x-axis and the second value controls the y-axis. The value can be a string (left, center or right), or a number (in px or %). Negative values are also allowed.

Using the object-position Property

Let's say that the part of the image that is shown, is not the part that we want. To position the image, we will use the object-position property.

Here we position the image so that the great old building is in center:

Example

```
.image-container {
  width: 200px;
  height: 300px;
  border: 1px solid black;
}

.image-container img {
  width: 100%;
  height: 100%;
  object-fit: cover;
  object-position: 80% 100%;
}
```

Here we will use the object-position property to position the image so that the famous Eiffel Tower is in center:

Example

```
.image-container {
  width: 200px;
  height: 300px;
  border: 1px solid black;
}

.image-container img {
  width: 100%;
  height: 100%;
```

```
  object-fit: cover;
  object-position: 15% 100%;
}
```
