If you have a CSS variable holding a color (e.g., a hex or `rgb()` value) and you want to apply an **alpha** (transparency) to it, you can use the modern **CSS Color Level 4** syntax with `rgb()` or `hsl()` and the `/` slash notation for alpha.

Example: Using a CSS variable with alpha

```
:root {
  --main-color: 34, 51, 170; /* store as RGB components */
}

.box {
  /* Apply alpha using slash syntax */
  background-color: rgb(var(--main-color) / 0.5); /* 50% opacity */
}
```

Why store as RGB components?

If you store the variable as #2233aa or `rgb(34, 51, 170)`, you can't directly inject alpha without parsing.
By storing only the numeric RGB values (`34, 51, 170`), you can reuse them in both opaque and transparent contexts.

Alternative: Using `hsl()`

```
:root {
  --main-hsl: 220 60% 50%;
}

.box {
```

```
  background-color: hsl(var(--main-hsl) / 0.3); /* 30% opacity */
}
```

If you already have a hex variable

CSS doesn't allow adding alpha directly to a hex stored in a variable like `--color: #2233aa;` in pure CSS.
You have two options:

1. **Change the variable format** to RGB/HSL components as above.
2. **Use** `color-mix()` (CSS Color 5, supported in modern browsers):

```
:root {
  --main-color: #2233aa;
}

.box {
  background-color: color-mix(in srgb, var(--main-color) 50%, transparent);
}
```

This blends the color with transparency.

✅ **Best practice:** Store colors as RGB or HSL components in variables if you need to frequently adjust alpha.

If you want, I can give you a **backwards-compatible solution** that works even in older browsers without `color-mix()`.
Do you want me to prepare that?