Grid Alignment Issue

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>CSS Filters</title>
    <style>
        body{
            display: flex;
            align-items: center;
            justify-content: center;
        }

        .containers{
            height: 100vh;
            width: 80vw;
            padding: 41px;
            box-sizing: border-box;
            display: grid;
            grid-template-columns: 1fr 1fr 1fr;
            background-color: antiquewhite;
        }
        .container1 img{
            height: auto;
            width: 100%;
            /* It becomes equal to the width of the grid column */
        }
    </style>
</head>
<body>
    <!-- Now, we have filter property in CSS using which we can control image filters like brightness, contrast, invert colors etc. and we can do much more

    Now, CSS filters is a basic concept and its a simple in-built feature-->
    <div class="containers">
```

```
        <div class="container1">
            <img src="https://en-media.thebetterindia.com/uploads/2024/11/Royal-Bengal-Tiger-1732281982.jpg" alt="Tiger Image">

        <div class="container1">
            <img src="https://en-media.thebetterindia.com/uploads/2024/11/Royal-Bengal-Tiger-1732281982.jpg" alt="Tiger Image">

        <div class="container1">
            <img src="https://en-media.thebetterindia.com/uploads/2024/11/Royal-Bengal-Tiger-1732281982.jpg" alt="Tiger Image">

        <div class="container1">
            <img src="https://en-media.thebetterindia.com/uploads/2024/11/Royal-Bengal-Tiger-1732281982.jpg" alt="Tiger Image">
        </div>
    </div>
</body>
</html>
```

here why did my image go vertical align even though its grid, I wanted to horizontal align them?

The issue you're facing—where the images are stacking vertically instead of aligning horizontally—is caused by incorrect **nesting** of your `.container1` `<div>` elements.

What's Wrong:

You've accidentally **nested multiple** `.container1` **divs inside each other** instead of placing them **side by side** within the `.containers` grid.

What's Happening:

```html
<div class="container1">
    <img ...>
    <div class="container1"> <!-- nested inside the above one -->
```

```
        <img ...>
        <div class="container1"> <!-- and so on -->
```

This causes the grid to treat the first `.container1` as a single item, and all the rest as its **nested children**, not as separate grid items. Hence, only one column is used, and everything appears vertically.

✅ Correct Version:

You should close each `.container1` properly **before** opening the next one. Like this:

```html
<div class="containers">
    <div class="container1">
        <img src="..." alt="Tiger Image">
    </div>
    <div class="container1">
        <img src="..." alt="Tiger Image">
    </div>
    <div class="container1">
        <img src="..." alt="Tiger Image">
    </div>
</div>
```

Bonus Tips:

- You used `grid-template-columns: 1fr 1fr 1fr;` — this will create 3 equal-width columns.
- With the corrected structure, your 3 images will align **horizontally**, one in each column.

Let me know if you want to apply different filters to each image too.

I fiexed thta, but I guess I accidendlty disvored a way to chnage the grid direction to vertical right? is it a good way or there are better alternatives for it because in flex we just have to chnage the flex-direction but here we have nothing

Great observation, Vaibhav! You're absolutely right — by **nesting elements**, you *accidentally* created a vertical stacking effect inside a grid. But here's the key point:

> **That is not the recommended way to make a CSS Grid layout vertical.**

✅ Correct Way to Make Grid Flow Vertically

CSS Grid **does** have a way to control the direction — it's done using the `grid-auto-flow` property.

🔹 By default:

```css
grid-auto-flow: row;
```

This means grid items fill **left-to-right, row-by-row**.

🔹 To make it vertical (column-wise filling):

```css
grid-auto-flow: column;
```

This will place grid items **top-to-bottom**, one below the other **in columns**, and then move to the next column when the first one is full.

🧪 Example:

```css
.containers {
    display: grid;
    grid-template-rows: repeat(3, 1fr); /* 3 rows */
    grid-auto-flow: column;             /* fills top to bottom */
    height: 100vh;
    width: 80vw;
    background-color: antiquewhite;
}
```

This layout will now stack items **vertically**, and once 3 rows are filled, it'll start a new column.

🆚 Flex vs Grid (Direction Control)

Property	Flexbox	CSS Grid
Default direction	`row`	`grid-auto-flow: row`
Change to vertical	`flex-direction: column`	`grid-auto-flow: column`
Best for	1D layouts	2D layouts

✅ Conclusion:

While your nesting trick *worked*, it's better to use:

- `grid-auto-flow: column` for vertical grid layouts.
- That keeps your HTML semantic and CSS readable.

Let me know if you want to visualize this with sample code or try auto-placement tricks!

↪ `grid-auto-flow: column` f

I rtried using it but still formed a very weierd ayout like 3 imsages were small and the last one of very big and they were horizontally centred only

You're absolutely right to expect `grid-auto-flow: column` to stack items vertically — but the weird behavior you're seeing (small images + one large image, horizontally centered) is likely due to **missing or inconsistent row heights and image sizing.**

Let me walk you through the correct way to create a vertical grid layout using `grid-auto-flow: column`, and how to fix what you saw:

✅ Correct CSS for Vertical Grid (Top to Bottom):

```html
```

```
<div class="containers">
    <div class="container1">
        <img src="..." alt="Tiger">
    </div>
    <div class="container1">
        <img src="..." alt="Tiger">
    </div>
    <div class="container1">
        <img src="..." alt="Tiger">
    </div>
    <div class="container1">
        <img src="..." alt="Tiger">
    </div>
</div>
```

```css
body {
    display: flex;
    align-items: center;
    justify-content: center;
    margin: 0;
}

.containers {
    display: grid;
    grid-auto-flow: column;          /* Fill top-to-bottom */
    grid-template-rows: repeat(3, 1fr); /* Three rows = 3 items per column */
    height: 90vh;
    width: 80vw;
    background-color: antiquewhite;
    gap: 20px;
}

.container1 img {
    width: 100%;       /* fill column width */
    height: auto;      /* preserve aspect ratio */
    display: block;
}
```

⚠️ What Was Causing the Weird Layout?

- You used `grid-auto-flow: column` but **didn't define** `grid-template-rows`, so the grid couldn't calculate how to stack properly.
- The last image appeared large because the grid stretched it to fill available vertical space (possibly one column instead of three rows).
- Without consistent `height` or row settings, the browser "guessed" how to place them — leading to a misaligned layout.

🔁 Optional Fix if You Want Fixed Image Size Instead:

```css
.container1 img {
    width: 100%;
    height: 200px; /* fixed height */
    object-fit: cover;
}
```

✅ In Summary

To create a **vertically flowing grid**:

1. Use `grid-auto-flow: column`.
2. **Explicitly define** `grid-template-rows` to control how many items per column.
3. Make sure child elements and images have consistent size behavior.

Let me know if you want to adapt it responsively or animate the layout!