CSS Notes
What is CSS?
CSS is an acronym stands for Cascading Style Sheets. It is a style sheet language which is used to describe the look and formatting of a document written in markup language. It provides an additional feature to HTML. It is generally used with HTML to change the style of web pages and user interfaces. CSS is used along with HTML and JavaScript in most websites to create user interfaces for web applications and user interfaces for many mobile applications
CSS is designed to make style sheets for the web. It is independent of HTML and can be used with any XML-based markup language. Now let's try to break the acronym:
Cascading: Falling of Styles
Style: Adding designs/Styling our HTML tags
Sheets: Writing our style in different documents
==========
CSS History
CSS 1 -:

The first CSS specification to become an official W3C Recommendation is CSS level 1, published on December 17, 1996. Hakon Wium Lie and Bert Bos are credited as the original developers.Among its capabilities are support for
Font properties such as typeface and emphasis
Color of text, backgrounds, and other elements
Text attributes such as spacing between words, letters, and lines of text
Alignment of text, images, tables and other elements
Margin, border, padding, and positioning for most elements
Unique identification and generic classification of groups of attributes
The W3C no longer maintains the CSS 1 Recommendation.

CSS 2 -:

CSS level 2 specification was developed by the W3C and published as a recommendation in May 1998. A superset of CSS 1, CSS 2 includes a number of new capabilities like absolute, relative, and fixed positioning of elements and z index, the concept of media types, support for aural style sheets (which were later replaced by the CSS 3 speech modules) and bidirectional text, and new font properties such as shadows.
The W3C no longer maintains the CSS 2 recommendation.
CSS 2.1 -:

CSS level 2 revision 1, often referred to as "CSS 2.1", fixes errors in CSS 2, removes poorly supported or not fully interoperable features and adds already implemented browser extensions to the specification. To comply with the W3C Process for standardizing technical specifications, CSS 2.1 went back and forth between Working Draft status and Candidate Recommendation status for many years. CSS 2.1 first became a Candidate Recommendation on February 25, 2004, but it was reverted to a Working Draft on June 13, 2005 for further review. It returned to Candidate Recommendation on 19 July 2007 and then updated twice in 2009. However, because changes and clarifications were made, it again went back to Last Call Working Draft on 7 December 2010.
CSS 2.1 went to Proposed Recommendation on 12 April 2011. After being reviewed by the W3C Advisory Committee, it was finally published as a W3C Recommendation on 7 June 2011.
CSS 2.1 was planned as the first and final revision of level 2—but low priority

work on CSS 2.2 began in 2015.
CSS 3-:

"CSS3" redirects here. For other uses, see CSS3 (disambiguation).
Unlike CSS 2, which is a large single specification defining various features, CSS 3 is divided into several separate documents called "modules". Each module adds new capabilities or extends features defined in CSS 2, preserving backward compatibility. Work on CSS level 3 started around the time of publication of the original CSS 2 recommendation. The earliest CSS 3 drafts were published in June 1999.

What are the uses of CSS?
CSS is used for defining the styles for web pages. It describes the look and formatting of a document which is written in a markup language. It provides an additional feature to HTML. It is generally used with HTML to change the style of web pages and user interfaces.
It is easier to make the web pages presentable using CSS. It is easy to learn and understand and used to control the presentation of an HTML document. CSS helps us to control the text color, font style, the spacing between paragraphs, sizing of columns, layout designs, and many more. It is independent of HTML, and we can use it with any XML-based markup language.It is recommended to use CSS because the HTML attributes are being deprecated. So, for making HTML pages compatible with future browsers, it is good to start using CSS in HTML pages.
There are several uses of CSS that are discussed as follows -:
1)Solves a big problem
Before CSS, tags like font, color, background style, element alignments, border, and size had to be repeated on every web page. This was a very long process. For example: If we are making a large website where fonts and color information are required to add on every page, it will be a long process. CSS was created to solve this problem. It was a W3C recommendation.
2)Saves a lot of time
CSS style definitions are saved in external CSS files, so it is possible to change the entire website by changing just one file.
3)Provide more attributes
CSS provides more detailed attributes than plain HTML to define the look and feel of the website.
4)Pages load faster
CSS does not require the writing of HTML tag attributes every time. There is the writing of rule just once for a tag, which can be applied to all the occurrences of the corresponding tag. So using CSS, there is less code, which means faster downloading.

5)Easier Website maintenance
CSS makes the maintenance of the website easier. It plays an essential role in website maintenance. If we require a global change in the file, it can be simply done by changing the style by which all the elements on the web page will update automatically. The CSS file provides a flexible look to the website, which can be altered in a convenient way. It also makes HTML formatting and modification of corresponding data elements easier.
6)Multiple device compatibility
CSS is compatible with the older language versions so that we can use CSS with the earlier language versions. Because of this, if the CSS application is developed with the older programming language versions and if the developer combines the same with new improvements, then CSS can be easily implemented with the

corresponding changes so the developer can update the existing code successfully. CSS allows the content to be optimized for more than one type of device.
Css 2
How To Write CSS?
There are 3 ways to write CSS in our HTML file-:
1.Inline CSS
2.Internal CSS
3.External CSS

1] Inline CSS -:
Are those css which are used with style attribute inside html element.
Before CSS this was the only way to apply styles
Not an efficient way to write as it has a lot of redundancy
Self-contained
Uniquely applied on each element
The idea of separation of concerns was lost

Syntax-:
Hello,Welcome to

2] Internal /Embedded CSS -:
Are those css which is used with <style> inside html document.
With the help of style tag, we can apply styles within the HTML file
Redundancy is removed
But the idea of separation of concerns still lost
Uniquely applied on a single document
Syntax-:
<style type="text/css">
p
{
color:red;
}
</style>
Write the above mentioned syntax in <head> tag.

3] External Css -:
Where we create external file having .css extension.
With the help of <link> tag in the head tag, we can apply styles
Reference is added
File saved with .css extension
Redundancy is removed
The idea of separation of concerns is maintained
Uniquely applied to each document

Syntax -:
p
{
color:red;
}
Save above the syntax in css file with .css extension & link css file inside HTML using <link> tag.

--Priority order--
Inline > Internal > External

If you apply all of these files the 1st priority will be given to Inline CSS then Internal CCC & then External CSS.

==========

CSS Selector

CSS selectors are used to select the content you want to style. Selectors are the part of CSS rule set. CSS selectors select HTML elements according to its id, class, type, attribute etc.

There are several different types of selectors in CSS.

1]CSS Element Selector
2]CSS Id Selector
3]CSS Class Selector
4]CSS Universal Selector
5]CSS Group Selector

1] CSS Element Selector -:

The element selector selects HTML elements based on the element name.

Syntax -:

```
<style>
p{
 text-align: center;
 color: blue;
}
</style>
```

2]CSS Id Selector -:

The id selector selects the id attribute of an HTML element to select a specific element. An id is always unique within the page so it is chosen to select a single, unique element.

It is written with the hash character (#), followed by the id of the element.

Syntax -:

```
<style>
#para1 {
 text-align: center;
 color: blue;
}
</style>
```

3]CSS Class Selector-:

The class selector selects HTML elements with a specific class attribute. It is used with a (.) character . (full stop symbol) followed by the class name.

Syntax -:

```
<style>
.center {
 text-align: center;
 color: blue;

}
</style>
```

=>If you want to specify that only one specific HTML element should be affected then you should use the element name with class selector.

```
<style>
p.center {
 text-align: center;
 color: blue;
}
```

</style>

4]CSS Universal Selector-:
The universal selector is used as a wildcard character. It selects all the elements on the pages.
Syntax-:
<style>
{
 color: green;
 font-size: 20px;
}
</style>

5]CSS Group Selector-:
The grouping selector is used to select all the elements with the same style definitions.
Grouping selector is used to minimize the code. Commas are used to separate each selector in grouping.
Syntax-:
h1,h2,p,#p1,.para {
 text-align: center;
 color: blue;
}

1]Font-:
Font Selection is Important.
Choosing the right font has a huge impact on how the readers experience a website.The right font can create a strong identity for your brand.Using a font that is easy to read is important. The font adds value to your text. It is also important to choose the correct color and text size for the font.
Generic Font Families
In CSS there are five generic font families:
1.Serif fonts have a small stroke at the edges of each letter. They create a sense of formality and elegance.
2.Sans-serif fonts have clean lines (no small strokes attached). They create a modern and minimalistic look.
3.Monospace fonts - here all the letters have the same fixed width. They create a mechanical look.
4.Cursive fonts imitate human handwriting.
5.Fantasy fonts are decorative/playful fonts.
Syntax-:
font-family: Arial, Helvetica, sans-serif;
font-size: medium | large | x-large | xx-large | xx-small | x-small | small | 20px;
font-style: Normal | italic | oblique;
font-variant: Normal | small-caps;
font-weight: normal | lighter | bolder | bold | number (100-900);
Defines from thin to thick characters. 400 is the same as normal, and 700 is the same as bold
==========
2]Background-:
The background property in CSS allows you to control the background of any element (what paints underneath the content in that element).
It is a shorthand property, which means that it allows you to write what would be multiple CSS properties in one.

Syntax-:
Background : (shorthand property)
background-color : colorName | HashCode | rgb() | rgba();
rgba : RED, GREEN, BLUE & ALPHA [for transparancy]
background-image : url();
background-repeat : no-repeat | repeat-x | repeat-y;
background-size:100% 100%;
background-position: top | center | bottom | left | top right | top center | top left | bottom right | bottom left | bottom center | center center ;
background-position: 25% 75%;
background-position: bottom 50px right 100px;
background-position: right 35% bottom 45%;
background-attachment: scroll | fixed;
==========
3]Border-:
The CSS border is a shorthand property used to set the border on an element.
The CSS border properties are use to specify the style, color and size of the border of an element. The CSS border properties are given below.
Syntax-:
border: 1px solid black;
(shorthand property)
border-bottom
border-top
border-left
border-right
border-style : solid | dashed | double | dotted | groove | ridge | inset | outset | hidden | none;
border-width: thin | medium | thick | 1px ;
border-color : red | rgb()| rgba() | hashcode ;

border-radius: 100%;
border-radius: top right bottom left;

--Outline--
CSS outline is just like CSS border property. It facilitates you to draw an extra border around an element to get visual attention.
It is as easy as to apply as a border.
Syntax-:
Outline: 1px solid black;
(shorthand property)
Outline-width:2px;
Outline-style: ; same as border given above
Outline-color:red | rgb() | hashcode ;
Outline-offset: 6px;
==========
4]Text-:
CSS has a lot of properties for formatting text.You can set following text properties of an element.
Syntax-:
color: colorname | Hashcode | rgb() , rgba();
text-align: center | left | right | justify;
text-decoration: overline | underline | line-through | none ;
text-transform: capitalize | uppercase | lowercase ;
text-indent: 40px ;
letter-spacing: 3px ;

word-spacing: 10px ;
line-height: 20px ;
==========

5]List-:
There are various CSS properties that can be used to control lists. Lists can be classified as ordered lists and unordered lists. In ordered lists, marking of the list items is with alphabet and numbers, whereas in unordered lists, the list items are marked using bullets.
Syntax-:
list-style-type: disc | circle | square | lower-alpha | upper-alpha | upper-roman | lower-roman | none ;
list-style-image: url(../Resources/close.png);
==========
6]Dimensions-:
The CSS height and width properties are used to set the height and width of an element.The height and width properties do not include padding, borders, or margins. It sets the height/width of the area inside the padding, border, and margin of the element.
Syntax-:
width: 20px | 100% | auto;
height: 20px | 100% | auto;
==========
7]Margin -:
The CSS margin properties are used to create space around elements, outside of any defined borders.With CSS, you have full control over the margins. There are properties for setting the margin for each side of an element (top, right, bottom, and left).Negative values are allowed.
All the margin properties can have the following values:
auto - the browser calculates the margin
length - specifies a margin in px, %, rem,em etc.
% - specifies a margin in % of the width of the containing element

Margin - Individual Sides
CSS has properties for specifying the margin for each side of an element:
Syntax-:
margin-top
margin-right
margin-bottom
margin-left
(Shorthand Property)
margin : top right bottom left;
margin : top&bottom right&left;
==========
8] Padding-:
The CSS padding properties are used to generate space around an element’s content, inside of any defined borders.With CSS, you have full control over the padding. There are properties for setting the padding for each side of an element (top, right, bottom, and left).Negative values are allowed.
All the margin properties can have the following values:
auto - the browser calculates the margin
length - specifies a margin in px, %, rem,em etc.
% - specifies a margin in % of the width of the containing element
Padding - Individual Sides
CSS has properties for specifying the padding for each side of an element:

Syntax-:
padding-top
padding-right
padding-bottom
padding-left
(Shorthand Property)
padding : top right bottom left;
padding : top&bottom right&left;
==========

9]Display-:
CSS display is the most important property of CSS which is used to control the layout of the element. It specifies how the element is displayed.
Every element has a default display value according to its nature.The default display value for most elements is block or inline.
Block-level Elements
A block-level element always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can).
Inline Elements
The inline element takes the required width only. It doesn't force the line break so the flow of text doesn't break in inline example.
Display: none;
display: none; is commonly used with JavaScript to hide and show elements without deleting and recreating them. Take a look at our last example on this page if you want to know how this can be achieved.
Syntax-:
Display: inline | block | inline-block | none;
==========
10]Position-:
The CSS position property is used to set position for an element. it is also used to place an element behind another and also useful for scripted animation effect.
You can position an element using the top, bottom, left and right properties. These properties can be used only after position property is set first. A position element's computed position property is relative, absolute, fixed or sticky.
There are five different position values:
1.static
2.relative
3.fixed
4.absolute
5.sticky
Elements are then positioned using the top, bottom, left, and right properties. However, these properties will not work unless the position property is set first. They also work differently depending on the position value.

Position: static;
HTML elements are positioned static by default.Static positioned elements are not affected by the top, bottom, left, and right properties.
An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page:
Position: relative;
An element with position: relative; is positioned relative to its normal position.
Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.
Position: fixed;

An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The top, right, bottom, and left properties are used to position the element.A fixed element does not leave a gap in the page where it would normally have been located.
Position: absolute;
An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed). However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.
Absolute positioned elements are removed from the normal flow, and can overlap elements.
Position: sticky;
An element with position: sticky; is positioned based on the user's scroll position.A sticky element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position:fixed).
Syntax-:
Position :static | relative | absolute | fixed | sticky ;
top:20px;
bottom:20px;
left:20px;
right:20px;

==========
11]Float-:
The CSS float property is a positioning property. It is used to push an element to the left or right, allowing other element to wrap around it. It is generally used with images and layouts.To understand its purpose and origin, let's take a look to its print display. In the print display, image is set into the page such that text wraps around it as needed.
Elements are floated only horizontally. So it is possible only to float elements left or right, not up or down.
1.A floated element may be moved as far to the left or the right as possible. Simply, it means that a floated element can display at extreme left or extreme right.
2.The elements after the floating element will flow around it.
3.The elements before the floating element will not be affected.
4.If the image floated to the right, the texts flow around it, to the left and if the image floated to the left, the text flows around it, to the right.
The float property can have one of the following values:
left - The element floats to the left of its container
right - The element floats to the right of its container
none - The element does not float (will be displayed just where it occurs in the text). This is default.
Synax-:
Float : left | right | none;

Clear
When we use the float property, and we want the next element below (not on right or left), we will have to use the clear property.The clear property specifies what should happen with the element that is next to a floating element.The clear property specifies on which sides of an element floating elements are not allowed to float.
The clear property can have one of the following values:

none - The element is not pushed below left or right floated elements. This is default.

left - The element is pushed below left floated elements
right - The element is pushed below right floated elements
both - The element is pushed below both left and right floated elements
When clearing floats, you should match the clear to the float: If an element is floated to the left, then you should clear to the left. Your floated element will continue to float, but the cleared element will appear below it on the web page.
Syntax-:
clear: none|left|right|both;
==========
12] Visibility-:
The CSS visibility property is used to specify whether an element is visible or not.An invisible element also take up the space on the page. By using display property you can create invisible elements that don't take up space.
CSS Visibility Parameters
visible:It is the by default value. It specifies that the element is visible.
hidden:It specifies that the element is invisible (but still takes up space).
Syntax-:
Visibility: visible | hidden ;
==========
13]Overflow-:
The CSS overflow property specifies how to handle the content when it overflows its block level container.We know that every single element on a page is a rectangular box and the size, positioning and behavior of these boxes are controlled via CSS.The overflow property specifies what should happen if content overflows an element's box.
This property specifies whether to clip content or to add scrollbars when an element's content is too big to fit in a specified area.
Let's take an example: If you don't set the height of the box, it will grow as large as the content. But if you set a specific height or width of the box and the content inside cannot fit then what will happen. The CSS overflow property is used to overcome this problem. It specifies whether to clip content, render scroll bars, or just display content.
visibleIt specifies that overflow is not clipped. it renders outside the element's box.this is a default value.

hidden It specifies that the overflow is clipped, and rest of the content will be invisible.
scroll It specifies that the overflow is clipped, and a scroll bar is used to see the rest of the content.
auto It specifies that if overflow is clipped, a scroll bar is needed to see the rest of the content.
inherit It inherits the property from its parent element.
initial It is used to set the property to its initial value.
Syntax-:
overflow-x
overflow-y
overflow:visible | scroll | hidden | auto;
==========
14] Cursor-:
It is used to define the type of mouse cursor when the mouse pointer is on the element. It allows us to specify the cursor type, which will be displayed to the user. When a user hovers on the link, then by default, the cursor transforms into

the hand from a pointer.
Syntax-:
cursor: alias | all-scroll | auto (default) | cell | context-menu | col-resize | copy
| crosshair | default | e-resize | ew-resize | grab | grabbing | help | move | n resize | ne-resize | nesw-resize
| ns-resize | nw-resize | nwse-resize | no-drop |
none | not-allowed | pointer | progress | row-resize | s-resize | se-resize | sw resize | text| vertical-text |
w-resize | wait | zoom-in | zoom-out ;
==========
15] Z-index-:
The z-index in CSS allows us to define a visual hierarchy on the 3-dimensional plane, i.e., the z-axis. It is used to specify the stacking order of the positioned elements (elements whose position value is either fixed, absolute, relative, or sticky). The stacking order means that the element's position along the z-axis, which is perpendicular to the screen.The z-index property specifies the stack order of an element.An element with greater stack order is always in front of an element with a lower stack order.
Note: z-index only works on positioned elements (position: absolute, position: relative, position: fixed, or position: sticky).
number: It means that the element's stack level is set to the given value. It allows negative values.
auto: It means that the order of the stack is equivalent to the parent, i.e., default.
Syntax-:
z-index: number | auto;
==========
16]Clip-path-:
This CSS property is used to create a clipping region and specifies the element's area that should be visible. The area inside the region will be visible, while the outside area is hidden. Anything outside the clipping will be clipped by browsers, including borders, text-shadows, and many more.It allows us to define a particular region of the element to display, instead of displaying the entire area. It makes it easier to clip the basic shapes by using ellipse, circle, polygon, or inset keywords.
Used to draw different shapes with using div.
Syntax-:
clip-path: polygon(100% 0, 50% 50%, 100% 100%, 41% 100%, 0 100%, 0 0);
==========
17]Pseudo-classes-:
Link-related pseudo class:
:link , :visited , :hover , :active
Input-related pseudo class:
:focus, :disabled, :checked, :required, :read-only, :valid,:invalid
Position/Number related pseudo class:
:root, :first-child, :last-child, :nth-child(), :empty
18]Pseudo-elements-:
Content-related:
::after (:after), ::before (:before)

Text related:
::first-letter, ::first-line, ::placeholder
Css 3
1]Border-:
The border-radius property defines the radius of the element's corners.This property allows you to add rounded corners to elements!
Syntax-:

border-radius : 1px 1px 1px 1px;
tl tr br bl ;
border-bottom-left-radius: 10px 70px;
border-bottom-right-radius: 50px 50px;
border-top-left-radius: 20px 200px;
border-top-right-radius: 20px 200px;
box-shadow:x axis y axis blur shadowColor;
==========
2]Text-:
The text-shadow property adds shadow to text.This property accepts a comma separated list of shadows to be applied to the text.
Syntax-:
text-shadow: x axis yaxis blur shadowColor;
writing-mode: horizontal-tb|vertical-rl|vertical-lr | sideways-rl | sideways-rl;
horizontal-tb -Let the content flow horizontally from left to right, vertically from top to bottom
vertical-rl -Let the content flow vertically from top to bottom, horizontally from right to left
vertical-lr -Let the content flow vertically from top to bottom, horizontally from

left to right
sideways-rl
sideways-lr
--Pointer Event--
Syntax-:
pointer-events: auto|none;
The pointer-events property defines whether or not an element reacts to pointer events.
==========
3]Transition-:
Syntax-:
transition-duration: 4s;
transition-timing-function: linear ,ease , ease-in ,ease-out , ease-in-out;
transition-delay: 1s;
==========
4]Transform-:
The transform property applies a 2D or 3D transformation to an element. This property allows you to rotate, scale, move, skew, etc., elements.
Syntax-:
translate() :It moves an object from one position to another
translateX() : Moves an object in x-axis
translateY(): Moves an object in y-axis
translate(x,y) : Moves an object in both axis
Note : values in pixel.
---
scale():It scales an element
scaleX()
scaleY()
scale(existing_width X value , existing_height Y value)
scale(x,y)
Note:Values In No.s.(Count).
---

rotate(): It rotates an object
rotateX()

rotateY()
rotateZ() : clockwise & anti clockwise rotates
Note:Values In Degree (40deg).
---
skew(x-axis, y-axis)
skewX(value in Degree)
skewY(value in Degree) : For Template
---
transform:rotateZ(90deg) translate(280px) scale(2);
==========
5]Animation-:
Syntax-:
animation-name : Name of the Animation
animation-duration :Duration of Animation
animation-delay : delay for that object
animation-iteration-count : number | infinite [loop through display or for particular time]
animation-direction:reverse|alternate|normal
animation [shorthand]
To create a animation using CSS first we need to create Frames
To create frames use "@keyframes & keyframe name"
Ex:

```
@keyframes moveIt
{
 from{css properties}
 to{css properties}
}

@keyframes moveIt
{
 0%{css properties}
 100%{css properties}
}
```

==========
6]Background-:
Syntax-:

```
background:linear-gradient(to bottom,red,orange,yellow);
background:radial-gradient(red,orange,yellow);
background:linear-gradient(to bottom,red,orange,yellow) , url();
```

==========
7]Display -:
This defines a flex container; inline or block depending on the given value. It enables a flex context for all its direct children.
Syntax-:
Property for parent elment -:
display: flex;
i. flex-direction Property
The flex-direction property specifies the direction of the flexible items.
Syntax-:
flex-direction: row (default) |row-reverse|column|column-reverse;
row (default): left to right in ltr; right to left in rtl
row-reverse: right to left in ltr; left to right in rtl
column: same as row but top to bottom
column-reverse: same as row-reverse but bottom to top

ii. flex-wrap Property
The flex-wrap property specifies whether the flexible items should wrap or not.
Syntax-:
flex-wrap: nowrap (default) | wrap | wrap-reverse;
iii. justify-content Property
The justify-content property aligns the flexible container's items when the items do not use all available space on the main-axis (horizontally).
Syntax-:
justify-content: flex-start (default) | flex-end | center | space-between | space-around | space-evenly;
flex-start (default): items are packed toward the start of the flex-direction.
flex-end: items are packed toward the end of the flex-direction.
start: items are packed toward the start of the writing-mode direction.
center: items are centered along the line
space-between: items are evenly distributed in the line; first item is on the start line, last item on the end line
space-around: items are evenly distributed in the line with equal space around them. Note that visually the spaces aren't equal, since all the items have equal space on both sides. The first item will have one unit of space against the container edge, but two units of space between the next item because that next item has its own spacing that applies.
space-evenly: items are distributed so that the spacing between any two items (and the space to the edges) is equal.
iv. align-items Property
The align-items property specifies the default alignment for items inside the flexible container.
Tip: Use the align-self property of each item to override the align-items property.
Syntax-:
align-items: stretch (default) | center | flex-start | flex-end | baseline;

Property for Child elment -:
v.flex-grow property
The flex-grow property specifies how much the item will grow relative to the rest of the flexible items inside the same container.

Syntax-:
flex-grow: number;
Default Value is 0.
vi.order Property
The order property specifies the order of a flexible item relative to the rest of the flexible items inside the same container.
Syntax-:
order: number;
Default Value is 0.
vii. align-self Property
The align-self property specifies the alignment for the selected item inside the flexible container.
Syntax-:
align-self: auto (default) | stretch | center | flex-start | flex-end | baseline;
==========
8]Filter-:
Syntax-:
filter: none | blur(5px) | brightness(100%) | contrast(200%) | drop-shadow(8px 8px 10px gray) | grayscale(100%) | hue-rotate(90deg) | invert(100%) | opacity(30%) | saturate(8) | sepia(100%) ;

The filter property defines visual effects (like blur and saturation) to an element (often <img>).

==========

9]Opacity-:

Syntax-:

opacity: number;

The opacity property sets the opacity level for an element.

The opacity-level describes the transparency-level, where 1 is not transparent at all, 0.5 is 50% see-through, and 0 is completely transparent.

==========

Thank You