Task2

1. What is CSS Flexbox & what are its properties ?

In CSS, we use Flexbox, a layout model that offers us a variety of ways to arrange the element and helps us align the element in CSS. With Flexbox's help, we can design responsive and dynamic webpage or user interface layouts. Flex containers are created by changing an element's display property to flex or inline-flex. The container is used as the parent element for the flex items and transforms into a flexible box or flex container. A flex container's direct child elements are transformed into flex items.

Properties

- **Flex Direction**:- With the help of flex-direction, we can arrange the element in any direction, like a row, a column, and a row reverse or a column reverse. This property is used to set the length of flexible items.
- **Flex Wrap**:- It specifies whether or not flex items should wrap across multiple lines or remain on a single line.
- **Flex Grow**:- The CSS flex-grow property specifies how much space a flex item should take if there is available space. The remaining space can be defined as the flex container size minus the size of all flex items together. This CSS property is used to increase the size of the flex-item.
- **Flex Shrink**:- The CSS flex-shrink property specifies how much an item will shrink than the other items of the container. It sets the flex shrink factor (a number that determines how much the flex item will shrink) of a flex-item.
- **Justify Content**:- Determines how flex items are aligned horizontally within the container using the justify-content property.
- **Align Items**:- The align- items property specifies the default alignment for items inside a flexbox or grid container. In a flexbox container, the flexbox items are aligned on the cross axis which is vertical by default. In a grid container the grid items are aligned in the block direction.

2. CSS position property & Associated values?

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.

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.

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.

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.

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).

Semantic Html Tag

A semantic element clearly describes its meaning to both the browser &developer.

Examples of non-semantic elements: & - Tells nothing about its content.

Examples of semantic elements: <form>,<table>,& <article> - Clearly defines its content.

<header>

The <header> element represents a container for introductory content or a set of navigational links.

A <header> element typically contains:

- one or more heading elements (<h1> - <h6>)
- logo or icon
- authorship information

<nav>

The <nav> tag defines a set of navigation links. The <nav> element is intended only for major blocks of navigation links.

<main>

The <main> tag specifies the main content of a document. The content inside the <main> element should be unique to the document. It should not contain any content that is repeated across documents such as sidebars, navigation links, copyright information, site logos, and search forms.

<section>

The <section> tag defines a section in a document.

<article>

The <article> tag specifies independent, self-contained content. An article should make sense on its own and it should be possible to distribute it independently from the rest of the site.

<aside>

The <aside> tag defines some content aside from the content it is placed in. The aside content should be indirectly related to the surrounding content.
The <aside> content is often placed as a sidebar in a document.

<footer>

The <footer> tag defines a footer for a document or section.

A <footer> element typically contains:

- authorship information
- copyright information
- contact information
- sitemap
- back to top links
- related documents