**Experiment No:** 09

**Experiment Name:** Introduction to CSS Flexbox and Grid.

**Theory:**

**Flexbox:** Flexbox is a one-dimensional layout model, primarily designed for arranging elements in a row or a column. It simplifies the creation of complex layouts where elements can dynamically adjust their size and position within a container, depending on available space.

**Display flex:** This defines a flex container; inline or block depending on the given value. It enables a flex context for all its direct children.

**Flex Direction:** This establishes the main-axis, thus defining the direction flex items are placed in the flex container.

**Justify Content:** This defines the alignment along the main axis. It helps distribute extra free space left over when all the flex items on a line are inflexible or flexible but have reached their maximum size.

**Align Items:** This defines the default behavior for how flex items are laid out along the cross-axis on the current line.

**Gap:** The gap property explicitly controls the space between flex items.

**Grid:** CSS Grid is a two-dimensional layout model that allows you to create grid structures consisting of rows and columns. It excels at creating complex, grid-based layouts, such as those found in magazines, newspapers, and responsive web applications.

**Grid Columns & Rows:** Defines the columns and rows of the grid with a space

separated list of values.

**Lab Task - 1(i):** Design web pages similar to the design given below using CSS flex and grid:

**i**



**Source Code:**

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Flexbox Layout</title>

    <link rel="stylesheet" href="flex1.css" />

  </head>

  <body>

    <div class="container">

      <div class="column column1"></div>

      <div class="column column2"></div>

      <div class="column column3"></div>

      <div class="column column4"></div>

    </div>

  </body>

</html>

Inside flex1.css:

body {

  display: flex;

  height: 100vh;

  background-color: #f0f0f0;

}

.container {

  display: flex;

  width: 50%;

  height: 300px;

  gap: 4px;

}

.column {

  flex: 1;

  height: 300%;

}

.column1 {

  background-color: hsl(158, 89%, 74%);

}

.column2 {

  background-color: #f56f7a;

}

.column3 {

  background-color: hsl(42, 95%, 63%);

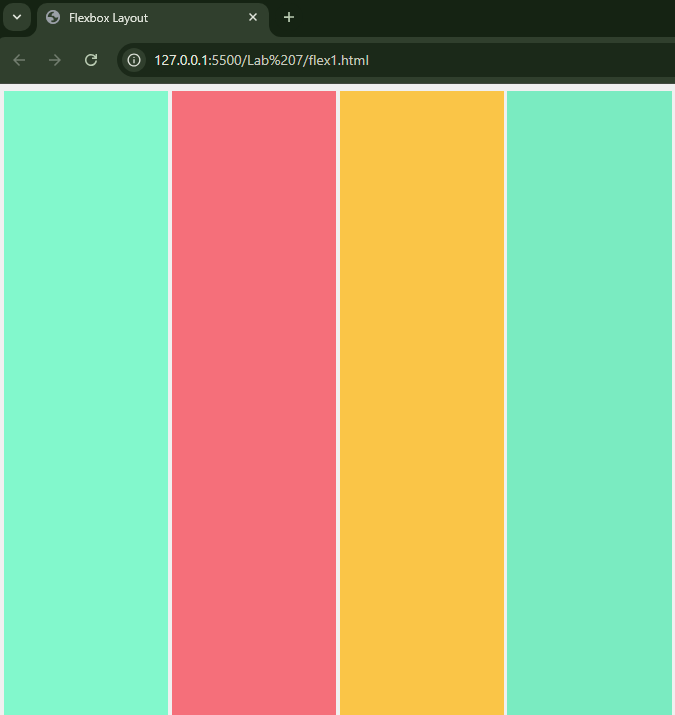
}

.column4 {

  background-color: rgb(122, 235, 193);

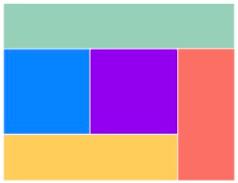
}

**Output:**

****

**Lab Task - 1(ii) :**Design web pages similar to the design given below using CSS flex and grid:

**ii**



**Source Code:**

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Flex and Grid Layout</title>

    <link rel="stylesheet" href="flex2.css" />

  </head>

  <body>

    <div class="container">

      <header class="header"></header>

      <aside class="aside"></aside>

      <main class="main"></main>

      <article class="article"></article>

      <footer class="footer"></footer>

    </div>

  </body>

</html>

Inside flex2.css:

body {

  margin: 0;

  display: flex;

  justify-content: center;

  align-items: center;

  height: 70vh;

}

.container {

  display: grid;

  grid-template-areas:

    "header header header"

    "aside main article"

    "footer footer article";

  width: 50%;

  height: 50vh;

}

header {

  grid-area: header;

  background-color: lightgreen;

  width: 73.3%;

}

aside {

  grid-area: aside;

  background-color: blue;

  height: 120%;

}

main {

  grid-area: main;

  background-color: purple;

  height: 120%;

}

article {

  grid-area: article;

  background-color: hsl(0, 73%, 51%);

  width: 150px;

}

footer {

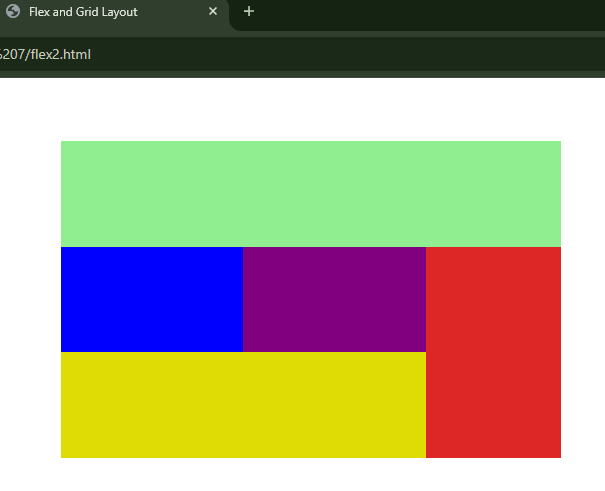
  grid-area: footer;

  background-color: rgb(223, 219, 5);

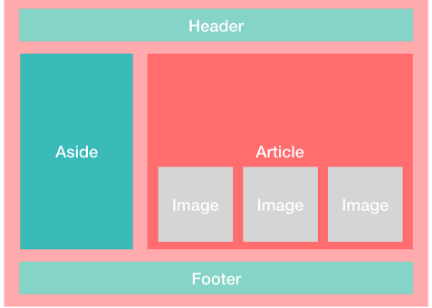
  height: 100%;

}

**Output:**

****

**Lab Task - 2:** Design a web page similar to this:



**Source Code:**

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>flexlayout</title>

    <link rel="stylesheet" href="flex3.css" />

  </head>

  <body>

    <div class="container">

      <div class="header">Header</div>

      <div class="main">

        <div class="aside">Aside</div>

        <div class="article">

          <div class="text">Article</div>

          <div class="imagess">

            <div class="image">Image</div>

            <div class="image">Image</div>

            <div class="image">Image</div>

          </div>

        </div>

      </div>

      <div class="footer">Footer</div>

    </div>

  </body>

</html>

Inside flex3.css:

\* {

  margin: 0;

  padding: 0;

  box-sizing: border-box;

}

body {

  display: flex;

  justify-content: center;

  align-items: center;

  height: 80vh;

  background-color: #f5c2c2;

}

.container {

  height: 500px;

  width: 700px;

}

.header {

  height: 60px;

  width: 675px;

  background-color: rgb(95, 207, 189);

  margin: 0 auto;

  margin-top: 12px;

  display: flex;

  justify-content: center;

  align-items: center;

  color: white;

  font-size: 26px;

  font-weight: 700;

}

.main {

  display: flex;

}

.aside {

  height: 330px;

  width: 210px;

  background-color: rgb(75, 153, 140);

  margin: 10px 0 0 12px;

  display: flex;

  justify-content: center;

  align-items: center;

  color: white;

  font-size: 20px;

  font-weight: 700;

}

.article {

  height: 330px;

  width: 450px;

  background-color: #d3584d;

  margin: 15px 0 0 15px;

  flex-direction: row;

  font-size: 20px;

  font-weight: 700;

}

.text {

  display: flex;

  justify-content: center;

  align-items: center;

  color: white;

  position: relative;

  top: 180px;

}

.imagess {

  display: flex;

  align-items: end;

  justify-content: space-evenly;

  position: relative;

  top: 190px;

}

.image {

  height: 100px;

  width: 110px;

  background-color: rgb(219, 217, 217);

  display: flex;

  align-items: center;

  color: white;

  justify-content: center;

  font-size: 20px;

  font-weight: 700;

}

.footer {

  height: 60px;

  width: 675px;

  background-color: rgb(95, 207, 189);

  margin: 0 auto;

  margin-top: 15px;

  display: flex;

  justify-content: center;

  align-items: center;

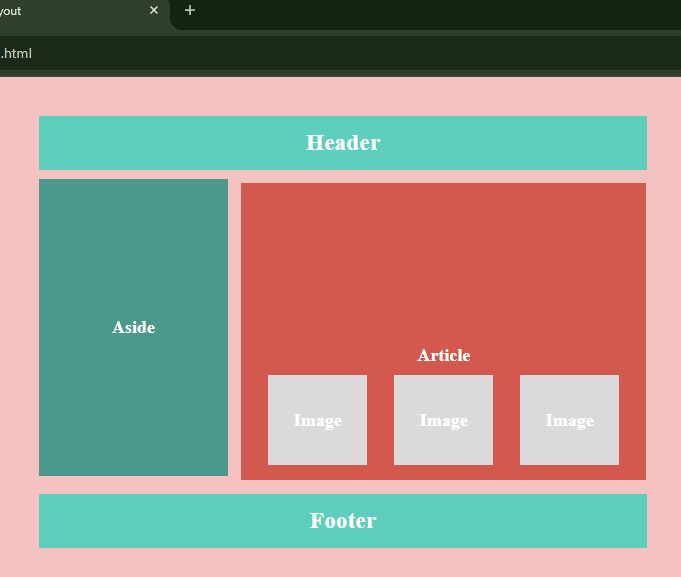
  color: white;

  font-size: 26px;

  font-weight: 700;

}

**Output:**

****

**Discussion:** In today’s lab, we got knew about CSS Flexbox and Grid. Flexbox is a one-dimensional layout model, primarily designed for arranging elements in a row or a column and CSS Grid is a two-dimensional layout model that allows you to create grid structures consisting of rows and columns. Flexbox simplifies the creation of complex layouts where elements can dynamically adjust their size and position within a container, depending on available space. Grid excels at creating complex grid-based layouts, such as those found in magazines, newspapers, and responsive web applications.By using flexbox and grid and their different attributes we completed our lab tasks.