**LAB MANUAL**

**| ARTIFICIAL INTELLEGENCE**

**Lab – 44**



A close up of a sign

Description automatically generated

**Building user Interface using HTML, CSS for ML/DL Models**

**Implementation of CSS:**

**Objective:**

The objective of this lab is to implement CSS concepts to style the Smart-Phone Price Prediction app. This lab will cover the following CSS components:

* Basic CSS properties and values
* Dynamic CSS3 features
* Box model
* Layout controls

**Equipment Required:**

1. Computer with internet access
2. Web Browser
3. VS Codre

**Prerequisites:**

1. Basic understanding of CSS
2. Familiarity with basic website development
3. Basic knowledge of using VS COde

**Problem Statement:**

Add CSS in smart phone prediction website with these concept:

* Basic CSS properties and values
* Dynamic CSS3 features
* Box model
* Layout controls

**Procedure:**

**1. Basic CSS Properties and Values**

**Objective:** Learn to use basic CSS properties and values to style HTML elements.

**CSS File: styles.css**

/\* Basic styling for the body and header \*/

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

background-color: #f0f0f0;

}

header {

background-color: #4CAF50;

color: white;

padding: 1em 0;

text-align: center;

}

**Explanation:**

* font-family: Sets the font family for the text.
* margin: Sets the margin outside the elements.
* padding: Sets the padding inside the elements.
* background-color: Sets the background color.
* color: Sets the text color.

**2. Dynamic CSS3 Features**

**Objective:** Implement dynamic CSS3 features like transitions, animations, and transformations.

**CSS File: styles.css**

/\* Adding transition effects to buttons \*/

input[type="submit"] {

background-color: #4CAF50;

color: white;

padding: 10px 20px;

border: none;

cursor: pointer;

margin-top: 10px;

transition: background-color 0.3s ease;

}

input[type="submit"]:hover {

background-color: #45a049;

}

**3. Box Model**

**Objective:** Understand and implement the box model properties.

**CSS File: styles.css**

/\* Styling the form and its elements using the box model \*/

form {

background-color: #ffffff;

padding: 20px;

border-radius: 5px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

margin: 20px auto;

max-width: 500px;

}

label {

display: inline-block;

width: 100px;

margin-bottom: 10px;

}

input[type="text"] {

width: calc(100% - 120px);

padding: 5px;

margin-bottom: 10px;

border: 1px solid #ccc;

border-radius: 4px;

}

**Explanation:**

* padding: Space inside the element's border.
* border: A line surrounding the element.
* margin: Space outside the element.
* box-shadow: Adds shadow effects around an element.
* border-radius: Rounds the corners of an element.

**4. Layout Controls**

**Objective:** Learn to control the layout of elements using CSS.

**CSS File: styles.css**

/\* Flexbox layout for the main content \*/

main {

display: flex;

flex-direction: column;

align-items: center;

padding: 20px;

}

/\* Grid layout for the table \*/

table {

width: 100%;

border-collapse: collapse;

margin: 20px 0;

display: grid;

grid-template-columns: repeat(8, 1fr);

}

th, td {

padding: 8px;

text-align: left;

border: 1px solid #ccc;

}

/\* Responsive design using media queries \*/

@media (max-width: 600px) {

table {

grid-template-columns: 1fr;

}

th, td {

display: block;

width: 100%;

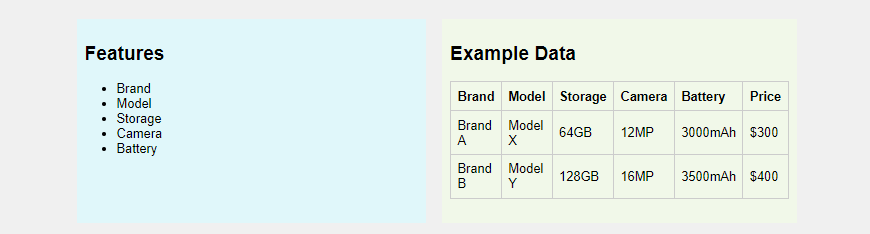
}

}

**Explanation:**

* display: flex: Creates a flexible container.
* flex-direction: Defines the direction of the flexible items.
* align-items: Aligns items along the cross axis.
* display: grid: Creates a grid container.
* grid-template-columns: Defines the number of columns in the grid.
* media queries: Adds responsive design to adapt to different screen sizes.

1. **Displaying list and table in two columns:**

****

/\* Flexbox layout for features and example-data sections \*/

.features, .example-data {

    flex: 1;

    padding: 10px;

}

.features {

    background-color: #e0f7fa;

}

.example-data {

    background-color: #f1f8e9;

}

/\* Container to hold features and example-data sections \*/

.container {

    display: flex;

    flex-wrap: wrap;

    margin: 20px auto;

    max-width: 900px;

    gap: 20px;

}

**The complete code:**

/\* Basic styling for the body and header \*/

body {

    font-family: Arial, sans-serif;

    margin: 0;

    padding: 0;

    background-color: #f0f0f0;

}

header, footer{

    background-color: #4CAF50;

    color: white;

    padding: 1em 0;

    text-align: center;

}

/\* Flexbox layout for features and example-data sections \*/

.features, .example-data {

    flex: 1;

    padding: 10px;

}

.features {

    background-color: #e0f7fa;

}

.example-data {

    background-color: #f1f8e9;

}

/\* Container to hold features and example-data sections \*/

.container {

    display: flex;

    flex-wrap: wrap;

    margin: 20px auto;

    max-width: 900px;

    gap: 20px;

}

/\* Adding transition effects to buttons \*/

input[type="submit"] {

    background-color: #4CAF50;

    color: white;

    padding: 10px 20px;

    border: none;

    cursor: pointer;

    margin-top: 10px;

    transition: background-color 0.3s ease;

}

input[type="submit"]:hover {

    background-color: #45a049;

}

/\* Styling the form and its elements using the box model \*/

form {

    background-color: #ffffff;

    padding: 20px;

    border-radius: 5px;

    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

    margin: 20px auto;

    max-width: 500px;

}

label {

    display: inline-block;

    width: 100px;

    margin-bottom: 10px;

}

input[type="text"] {

    width: calc(100% - 120px);

    padding: 5px;

    margin-bottom: 10px;

    border: 1px solid #ccc;

    border-radius: 4px;

}

/\* Flexbox layout for the main content \*/

main {

    display: flex;

    flex-direction: column;

    align-items: center;

    padding: 20px;

}

/\* Grid layout for the table \*/

table {

    width: 100%;

    border-collapse: collapse;

    margin: 20px 0;

    display: grid;

    grid-template-columns: repeat(8, 1fr);

}

th, td {

    padding: 8px;

    text-align: left;

    border: 1px solid #ccc;

}

/\* Responsive design using media queries \*/

@media (max-width: 600px) {

    table {

        grid-template-columns: 1fr;

    }

    th, td {

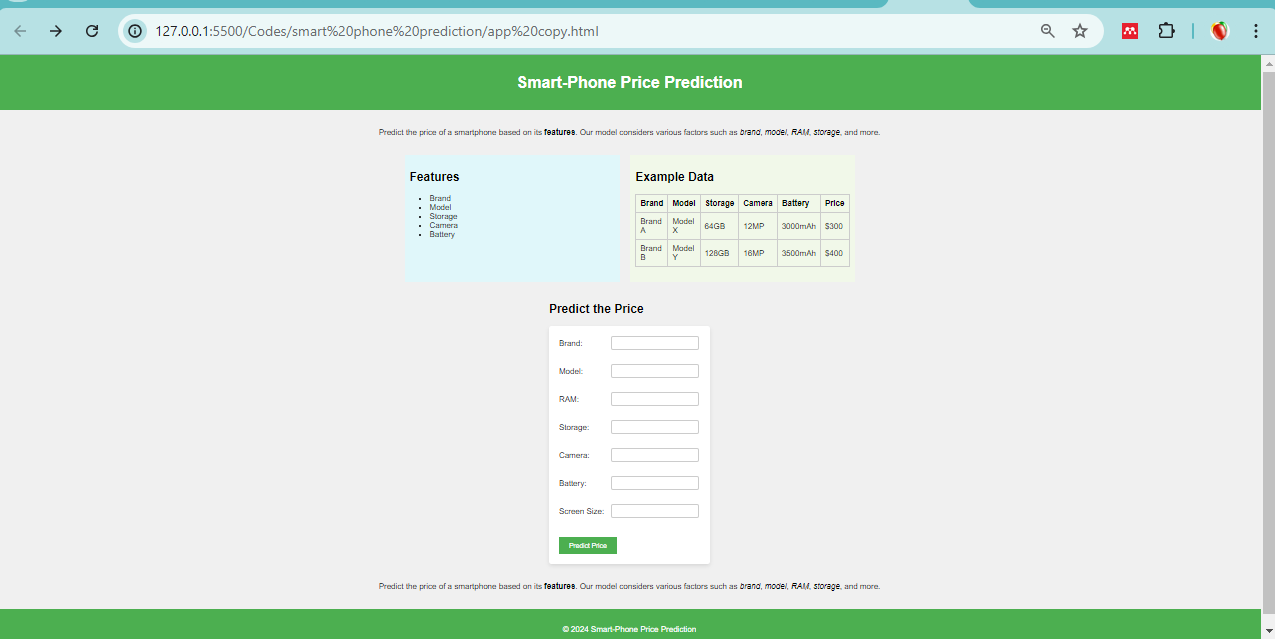
        display: block;

        width: 100%;

    }

}

**Output:**

****

**Conclusion**

By following this lab manual, you should be able to style a simple web application using CSS. This exercise will help you understand the basics of CSS, including basic properties, dynamic CSS3 features, the box model, and layout controls.