

# report2

## 布局与定位

姓名	学号	班级	实验时间
张晓琳	423109071102	计科2班	2025-11-11

### 实验环境说明

- 1. 操作系统： Windows 11
- 2. 浏览器： Chrome 142.0
- 3. 编辑器： VS Code

### 实验过程描述

#### 流体布局 (float + clear)

##### 操作步骤简述：

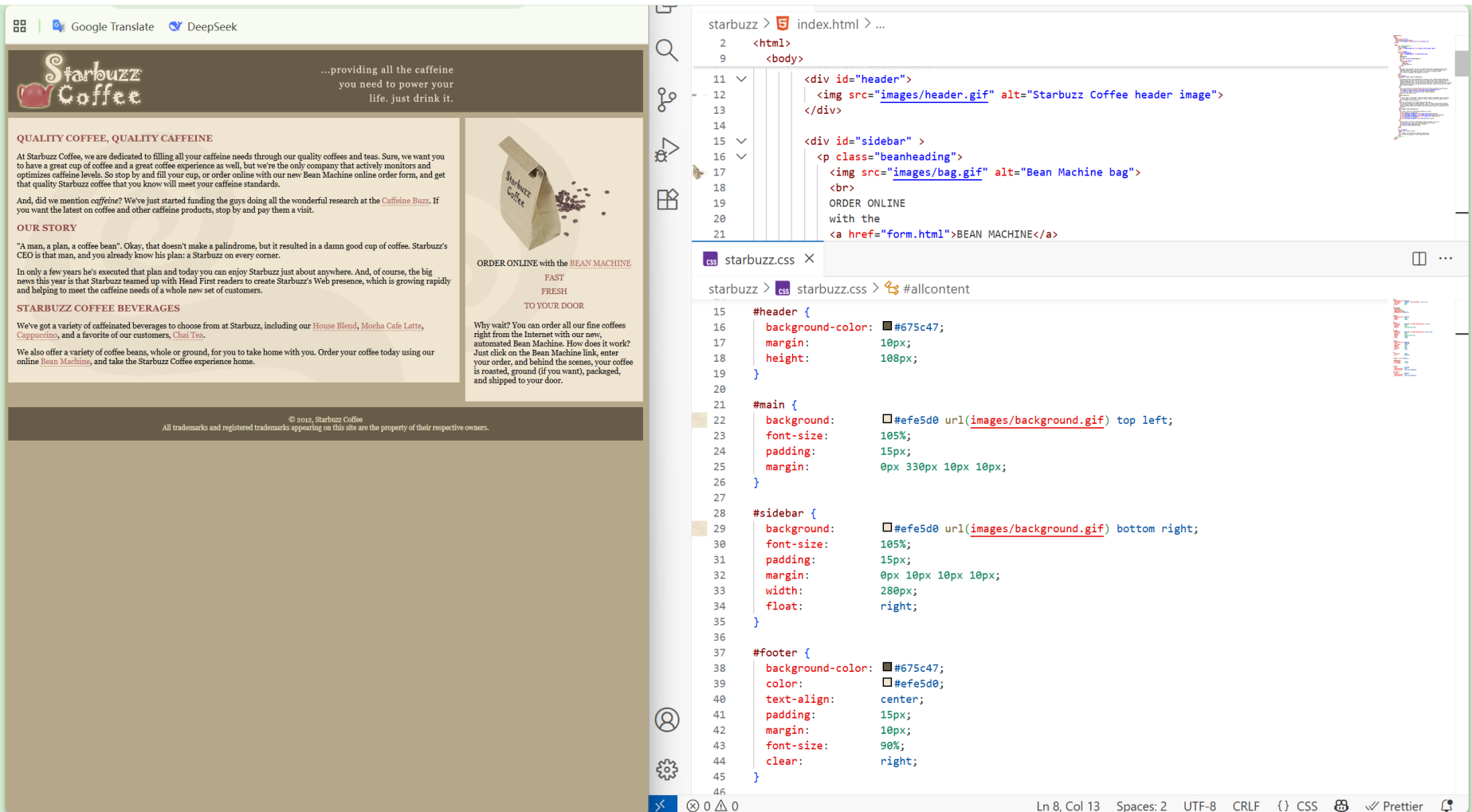
首先将 `<div id="sidebar">` 移动到 `<div id="header">` 下方，然后为**sidebar**设置 `width: 280px` 和 `float: right` 属性，使侧边栏向右浮动。接着为**main**区域设置 `margin-right: 330px`，为**footer**设置 `clear: right` 清除浮动。

##### 预期效果与实际效果：

预期侧边栏会浮动在右侧，主内容环绕其左侧，页脚清除浮动位于底部。实际效果符合预期，实现了经典的两栏流体布局，主内容区域会自适应剩余空间。

##### 遇到的问题与解决方法：

最初忘记为**footer**设置**clear**属性，导致页脚上浮到侧边栏旁边。通过添加 `clear: right` 成功解决。



## 冻结布局 (fixed width container)

### 操作步骤简述：

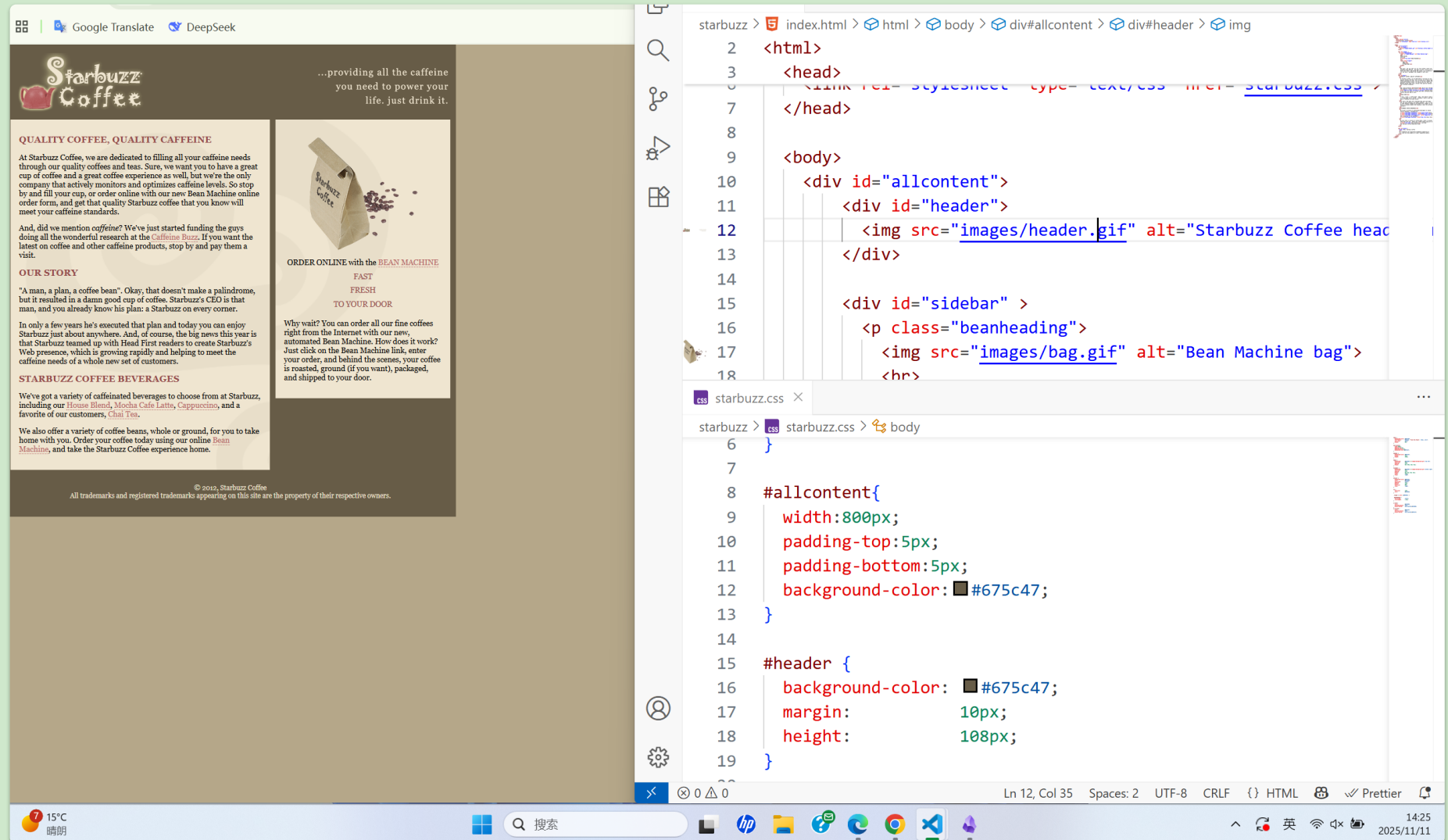
添加新的 `<div id="allcontent">` 容器包裹所有内容，为其设置固定宽度 `width: 800px` 和内边距，并设置背景色。

### 预期效果与实际效果：

预期整个内容区域会被固定为800px宽度，不再随浏览器窗口大小变化。实际效果良好，页面在窄窗口中会出现水平滚动条，布局更加稳定。

## 遇到的问题与解决方法：

无显著问题，实现过程较为直接。



## 凝胶布局 (margin auto)

### 操作步骤简述：

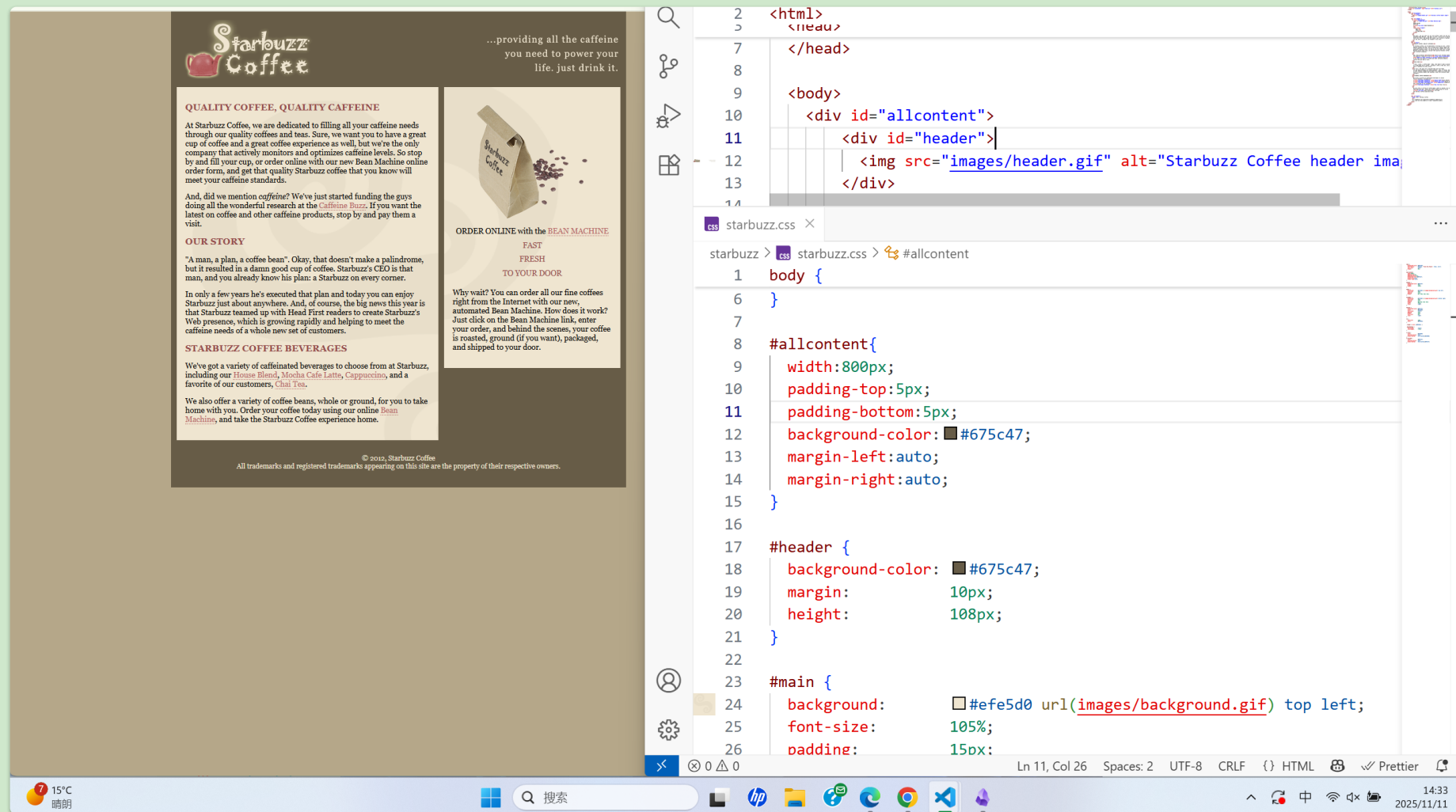
在冻结布局的基础上，为allcontent容器添加 margin-left: auto 和 margin-right: auto，使其水平居中。

## 预期效果与实际效果：

预期固定宽度的容器会在浏览器中水平居中显示。实际效果完美，页面在大屏显示器中居中显示，提升了视觉体验。

## 遇到的问题与解决方法：

无问题，这是实现水平居中的标准方法。



## 绝对定位布局 (position: absolute)

### 操作步骤简述：

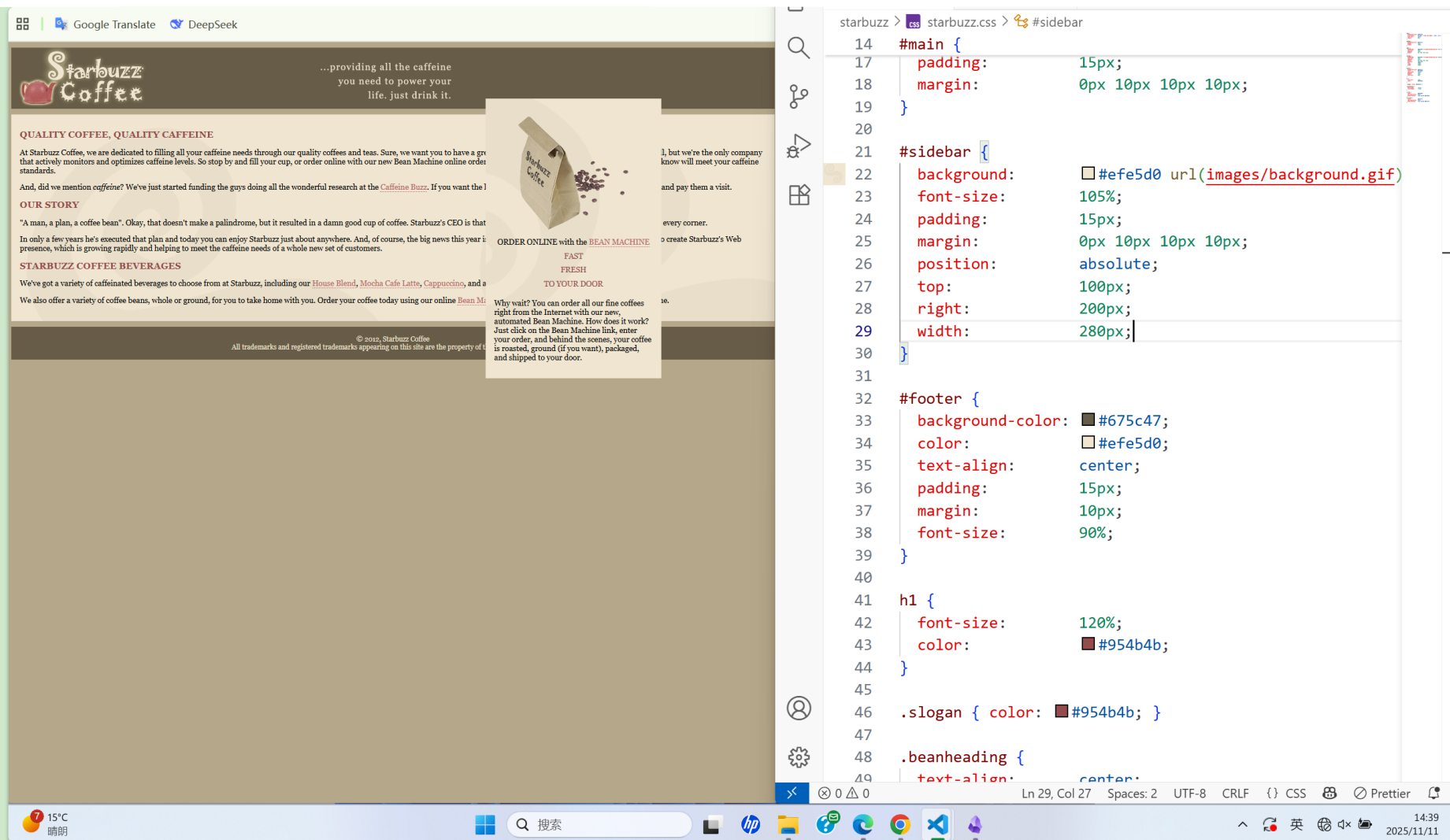
将页面恢复到初始状态，为**sidebar**设置 `position: absolute`、`top: 100px`、`right: 200px` 和 `width: 280px`。

### 预期效果与实际效果：

预期侧边栏会脱离文档流，固定在距离顶部**100px**、右侧**200px**的位置。实际效果符合预期，但注意到绝对定位元素不会影响其他元素的布局。

### 遇到的问题与解决方法：

发现主内容区域与侧边栏有重叠，这是绝对定位的固有特性，需要通过调整其他元素的外边距来解决。



## CSS 表格显示布局 (display: table, table-row, table-cell)

### 操作步骤简述:

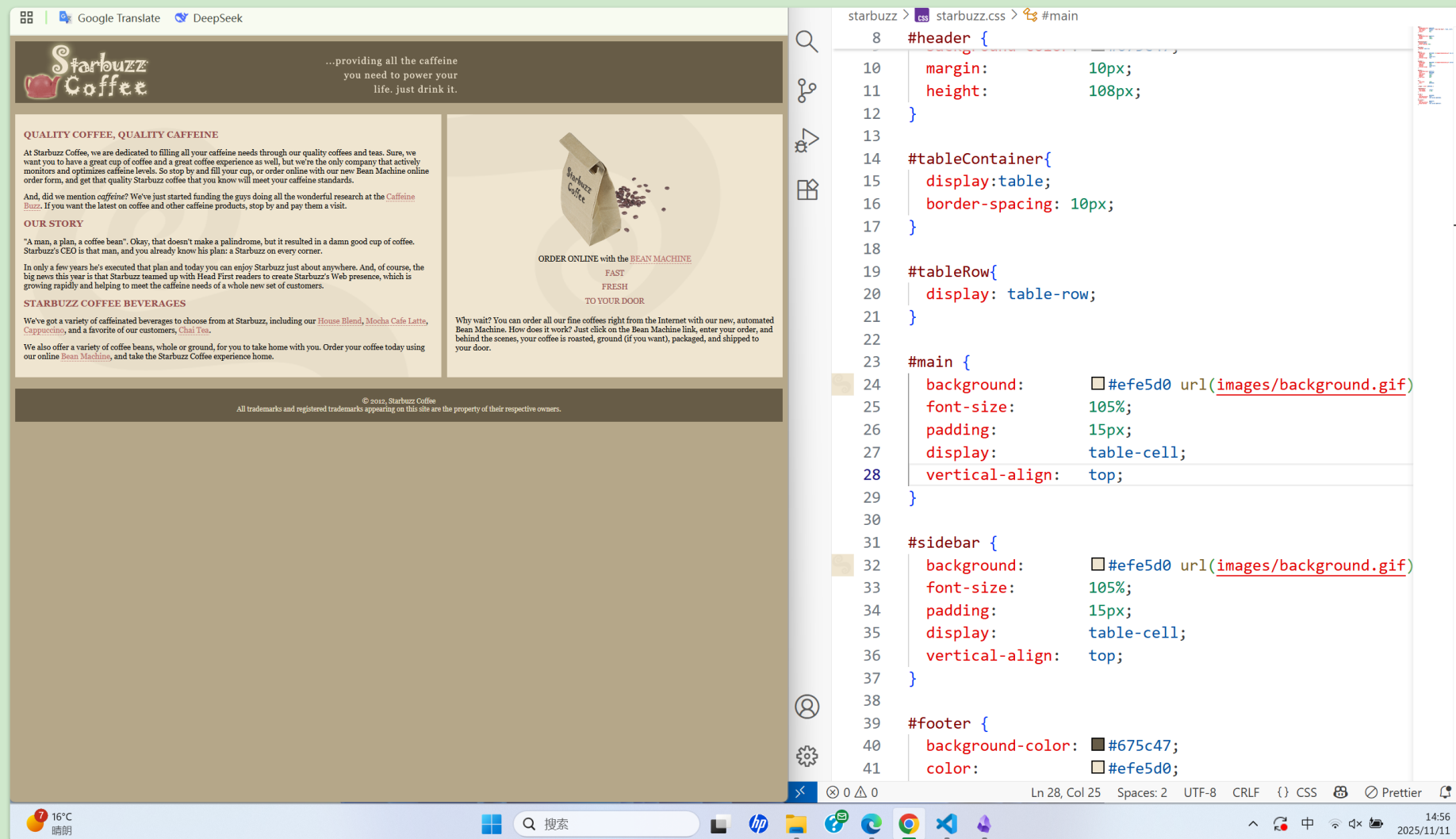
添加

## 预期效果与实际效果：

预期两个栏目会像表格单元格一样等高排列。实际效果很好，两个栏目自动等高，布局整齐美观。

## 遇到的问题与解决方法：

最初忘记设置 `vertical-align: top`，导致内容基线对齐不美观。添加该属性后问题解决。



## Flex 布局 (display: flex)

### 操作步骤简述：

添加**container**容器包裹**main**和**sidebar**，设置 `display: flex` 和 `flex-direction: row`。然后为**main**设置 `flex: 2 1 200px`，**sidebar**设置 `flex: 1 0.5 100px`。

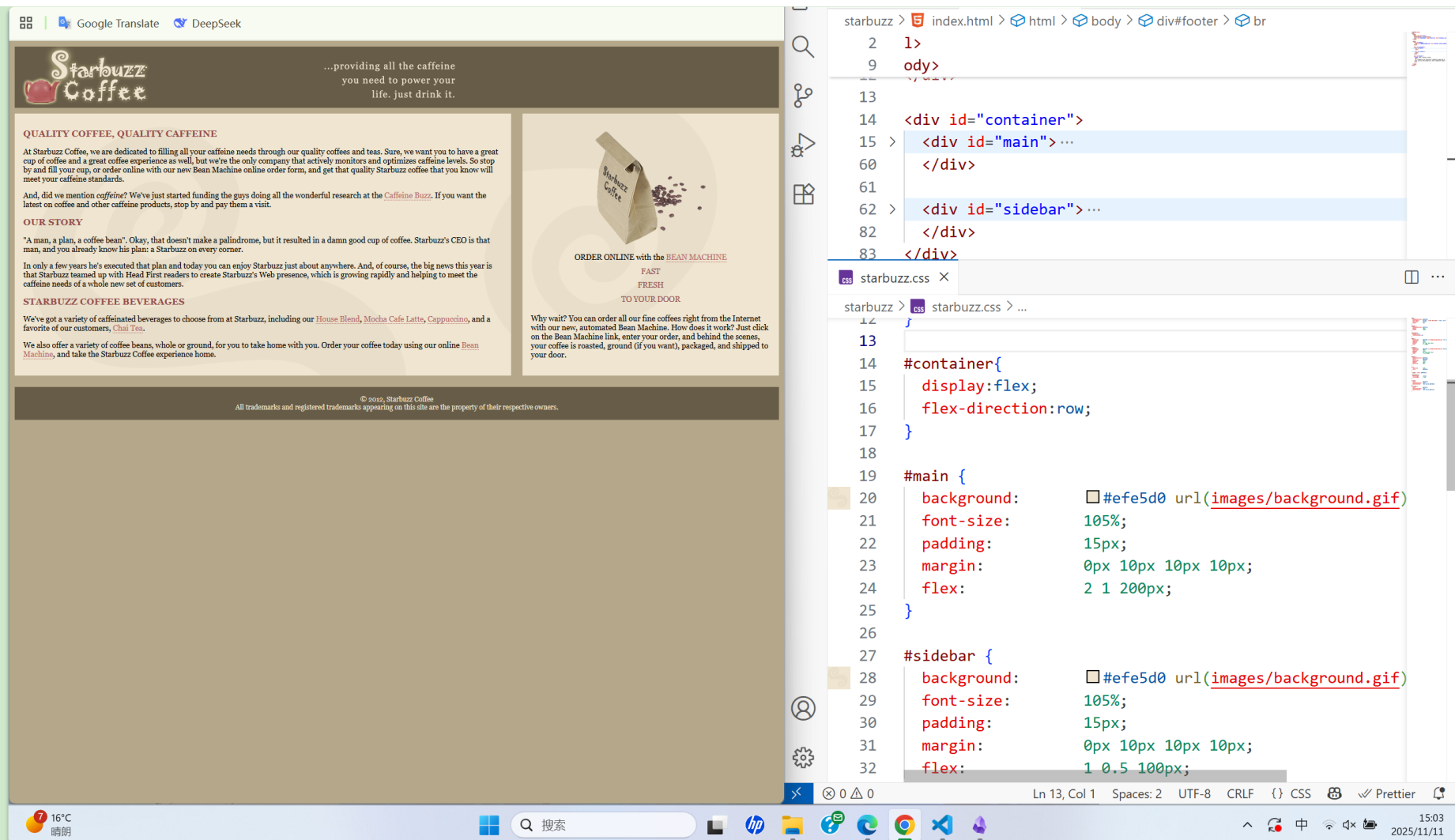
### 预期效果与实际效果：

预期两个栏目会按**1:2**的比例排列，具有灵活的伸缩性。实际效果优秀，布局响应性好，在不同屏幕尺寸下都能保持良好的比例关系。

### 遇到的问题与解决方法：

初次设置**flex**属性时对三个参数理解不够，通过查阅文档明确了**flex-grow**、**flex-shrink**和**flex-basis**的含义。





## 总结与感想

通过本次实验，我对**CSS**布局技术有了更深入的理解。每种布局方式都有其独特的优势和适用场景：流体布局适合需要自适应的页面，冻结和凝胶布局提供更稳定的视觉体验，绝对定位适合需要精确定位的元素，**CSS**表格显示能轻松实现等高布局，而**Flex**布局则是现代响应式设计的首选。

在实际开发中，我会根据具体需求选择合适的布局方式。对于简单的两栏布局，**float**仍是不错的选择；对于需要精确控制的布局，绝对定位很有用；而对于复杂的多栏响应式布局，**Flexbox**和**Grid**是更好的选择。特别是**Flex**布局，它的灵活性和强大的对齐能力让我印象深刻，已经成为现代网页布局的核心技术。

这次实验让我认识到，掌握多种布局技术并理解其原理至关重要。在实际项目中，往往需要结合使用多种技术才能达到最佳效果。同时，浏览器的开发者工具在调试布局问题时发挥了巨大作用，能够实时查看和修改样式，大大提高了开发效率。