**THIẾT KẾ MENU**

Để tạo menu trước tiên tạo 1 khối div để chứa menu đó

* Để Div hiển thị cần chỉnh màu, height để làm xuất hiện, nd đã có thì k cần height
* Khi tạo sẽ không khít màn hình vì thẻ body, html có margin nên ta phải css margin đó.
* Css 1 mình body cũng dc vì html trùng với body, k thể css 1 mình html. Nên css cả 2

Tiếp theo, thành phần của menu là danh sách nên ta tạo ul trong div

* Khi tạo thì k hiểu sao 1 phần ul nằm trên div, ul có margin: 16px 0px, padding left: 40px
* Nên ta phải css margin: 0 16px vì nằm trong div và cách div 16px. Trên mạng chỉnh margin : auto 16px cũng tương tự và chỉnh padding left : 0 để nội dung sát với viền
* Để thấy ul nằm đâu, ta ghi nd (li hoặc chữ) vào thẻ ul
* li nằm trùng với ul, nhưng chỉ mục lại nằm ở ngoài ul ( nằm ở margin)
* Vì li là giống nhau nhưng khi muốn li khác với những li còn lại (css) ta phải tạo ul khác

Nằm ngang

* Nằm ngang thì có display: block/ inline-block (css ở chính nó, không liên quan đến lớp cha) hoặc display: flex ( css ở lớp cha của nó, có 2 hình hcn bao quanh item)
* Có sự khác biệt ở đây vì block/ inline-block sẽ chiếm đủ dài và rộng còn flex sẽ chiếm hết chiều dài của container (lớp cha), làm cho item ở trung tâm (điều chỉnh lại margin của item)
* Các item có block/ inline-block sẽ cách nhau 1 đoạn còn flex sẽ cách nhau 1 khoảng lớn bằng ô vuông (do justify-content, do khoảng cách của 2 ul) ????

Chỉnh li nằm ngang: display : block/inline-block

* Ta thấy li đang chiếm hết chiều ngang ( vì display mặc định : block)
* Vì thế ta phải chỉnh các li nằm cùng 1 hàng bằng css display : inline / inline-block, khi đó mục đầu dòng sẽ bị mất
* List-style (lms lab9) (short-hand của type, position, image) : none sẽ làm mục đầu dòng bị mất đi nhưng khi chỉnh display như trên thì đã mất sẵn rồi.
* Các li quá gần nhau nên ta margin : 0 8px

Chỉnh ul nằm ngang: display : flex

* Khi đó div là container (chứa display : flex) và ul là items, còn li k phải
* Vì sao nằm ngang, vì có thuộc tính mặc định flex-direction : row mặc dù ch gõ

Để căn chỉnh item ul ta dùng: Justify-content

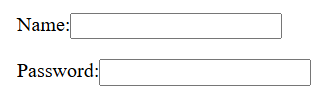
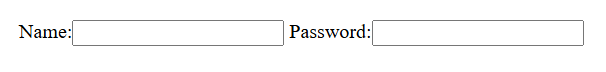
* Justify-content : flex start (mặc định) làm cho nằm ở đầu trục chính nên ta css lại Justify-content : space-between

**THIẾT KẾ FORM INPUT**

Mong muốn các thẻ lable, input bằng nhau và chiếm hết không gian còn lại (lớp cha)

Để tạo form input ta phải tạo 1 div chứ form đó:

* Vì không có lớp div bao phủ div form nên muốn div form hiển thị ở giữa màn hình, ta phải chỉnh width, margin, và nhờ auto (margin : 36px auto 0 auto) vì chiều rộng không đủ màn hình, sẽ tự động auto canh giữa trái phải
* Để nội dung cách ngoài viền form ta dùng padding
* Vì ta thấy ở đây nếu div lớn là container thì item sẽ là 2 div nhỏ thì k thể css lable, input mà chỉ css dc div. Vậy ở đây có 2 container là 2 div nhỏ chứ lable, input
* Vì thế ta css cho form\_row display: flex
* Ở đây ta ko thấy khoảng cách của các item, đều nằm sát trái (justify-content)
* Khi ta chỉnh display : inline- flex cho form\_row vì có 2 div chung class form\_row nên 2 div này sẽ hiển thị trên cùng 1 hàng ngang

Flex Inline-flex

CSS width cho lable, input bằng nhau

* Nếu để lable, input = nhau thì ta có thể chỉnh width, nhưng ở đây không được vì kích thước lable đã cố định ( khi chưa có flex), nhưng input thì được
* Vì thế sau khi display: flex thì width/ min-width có tác dụng nhưng max-width k

CSS cho width input chiếm hết không gian của lớp cha

* Trong layout flex có nhiều item, chỉ có 1 item có flex > 1 thì chiếm hết không gian có thể chiếm theo chiều của trục main axis

**Bố cục Website SLIDEBAR & MAIN**

**(Column\_Layout)**

Bình thường khi làm website, bên trái để sidebar thanh điều hướng, giữa làm nội dung, bên trái để sidebar quảng cáo. Sau đó chỉnh kích thước của từng phần

Tạo div chứa 3 phần gồm 2 sidabar trái phải và 1 main:

* Vì ta thấy item ở đây là các phần nên ta tạo 1 div bao phủ nó để có display flex
* Mỗi phần là 1 div, do đó item ở đây là div
* Khi tạo xog ta thấy margin-top của h2 sẽ nằm ngoài div, làm cho div có khoảng cách với phần tử khác và h2 không thể cách div 1 đoạn. Để làm điều này phải css padding của div. Khi đó khoảng cách h2 và div = padding + margin h2

Cho các item nằm ngang: display: flex

* Ta css ở div cha
* Sau đó, ta thấy kích thước mỗi phần chưa hợp lý. Kích thước này dựa vào nội dung bên trong

Chỉnh kích thước từng phần:

* Vì mỗi phần đều sử dụng 1 class, nếu ta muốn chỉnh kích từng mỗi như nhau thì ta css class đó flex > 1. Tất cả item có flex = 1 thì sẽ chia đều
* Ta thấy kích thước sẽ do flex. Vì thế khi muốn chỉnh phần nào lớn hơn thì nhân flex tương ứng
* Khi thu nhỏ màn hình thì giữ nguyên tỉ lệ đó

**Landing Page**

* Khi div cha chồng div con thứ 1 thì margin sẽ chồng lên nhau. Margin con k ảnh hưởng lên cha
* Để chỉnh kích thước bằng nhau ta có flex > 1 và khoảng trắng giữa các item thì dùng margin nhưng còn 1 cách
* Flex-basis để kích thước với chiều trùng với chiều của main axis. Vì thế ta có thể đưa giá trị vào flex-basis để có kích thước bằng nhau, nhưng phải chừa ra 1 chút kích thước cửa sổ tầm nhìn để sử dụng justify-content để tạo khoảng trống
* Justify-content: space-between, space-around, space-evenly đều tạo khoảng trống giữa các item nhưng khác nhau space around có khoảng trống 2 bên. Space-evenly giống với space – around nhưng khác ở chỗ space-evenly khoảng trống 2 bên cùng kích thước với khoảng trống item còn around thì khoảng cách 2 bên bằng 1 nửa

**Wrap Layout:**

Chú ý về font short hand bắt buộc có font-size và font-family

Để canh giữa chữ thì nhờ line-height và text-align

Xuống dòng flex-wrap: wrap/ wrap-reverse

**Center Item:**

Cách chỉnh item center: display: flex, margin auto

display: flex, justify-content : center (trục main axis)

align-item : center (trục cross axis)

**Chart Layout**

Kỹ thuật đặt biến --variable cục bộ và sử dụng biến height: var(--variable)

Khai báo biến trong thuộc tính style của thẻ (inline), biến đó sẽ là biến local, khi đó biến chỉ có thể được sử dụng với thẻ chứa nó

Kết hợp giữa inline và internal. Nếu có nhiều thẻ mang class chung thì inline là thuộc tính riêng của từng thẻ và internal chứa thuộc tính chung. Inline chứa biến thì internal sử dụng biến thì mỗi thẻ sẽ có giá trị biến khác nhau

Khi sử dụng flex thì nó hiển thị theo chiều ngang và cột div hướng xuống dưới vì mặc định nó sẽ tập trung về cross-start của trục cross axis với thuộc tính align-item: flex-start (mặc định) thể hiện điều đó. Nên ta phải css align-item : flex-end

@Keyframes

* Trong CSS, `@keyframes` là một quy tắc được sử dụng để định nghĩa các bước chuyển đổi (hoặc khung hình) của một \*\*animation\*\*.
* \*\*. Nó cho phép bạn tạo ra các hiệu ứng chuyển động mượt mà bằng cách xác định các trạng thái khác nhau của một thuộc tính CSS tại các thời điểm cụ thể trong thời gian chạy của một quá trình hoạt ảnh (animation).
* ### Cách làm việc của `@keyframes`:

- `@keyframes` định nghĩa cách các thuộc tính CSS thay đổi trong suốt thời gian của một \*\*animation\*\*.

- Cụ thể, bạn cần định nghĩa một cái tên cho animation đó và cài đặt các bước chuyển đổi ở các mốc phần trăm (hoặc sử dụng `from` và `to` để biểu thị điểm bắt đầu và điểm kết thúc).

* - Bạn có thể chia hoạt ảnh ra làm nhiều khung bằng cách sử dụng các mốc thời gian như 10%, 20%, 50%, v.v.
* - `from` thực chất tương đương với `0%` (bắt đầu) và `to` tương đương với `100%` (kết thúc).
* ### Cấu trúc cú pháp cơ bản của `@keyframes`:

```css

@keyframes ten-animation {

/\* Điểm bắt đầu \*/

from {

/\* Các thuộc tính CSS tại điểm bắt đầu \*/

}

/\* Điểm kết thúc \*/

to {

/\* Các thuộc tính CSS tại điểm kết thúc \*/

}

/\* Hoặc bạn có thể dùng phần trăm để định nghĩa \*/

0% {

/\* Trạng thái ban đầu \*/

}

50% {

/\* Trạng thái ở giữa \*/

}

100% {

/\* Trạng thái kết thúc \*/

}

}