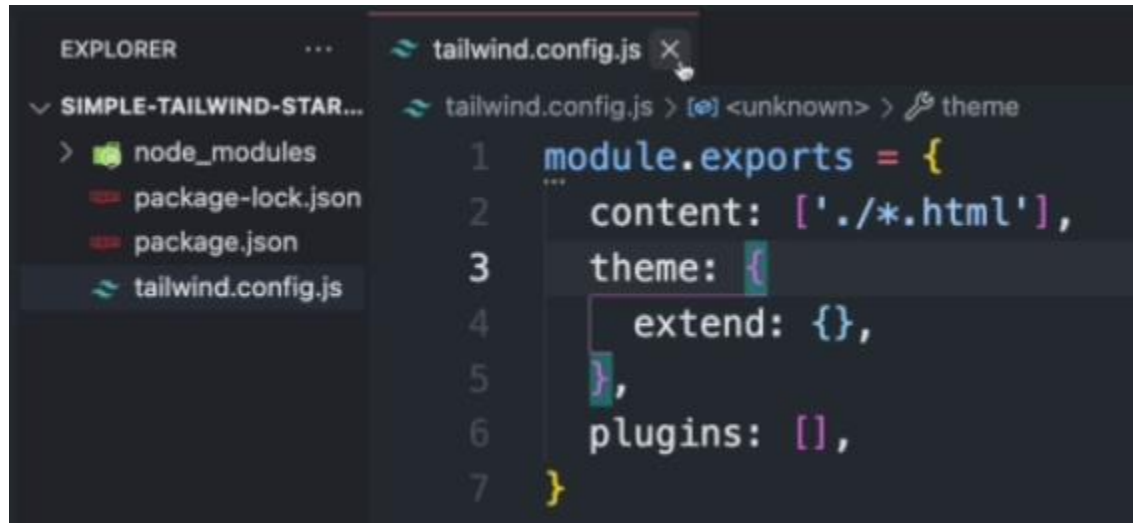
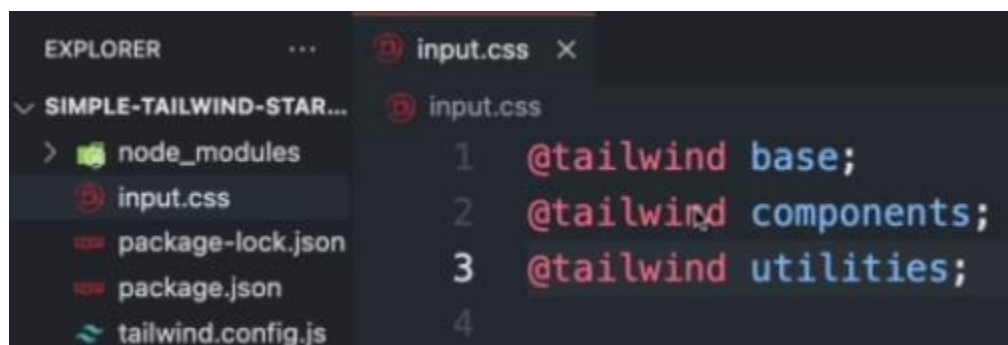


Create An Environment with Tailwind CLI :

Create a folder for project and follow te tailwind instructions.



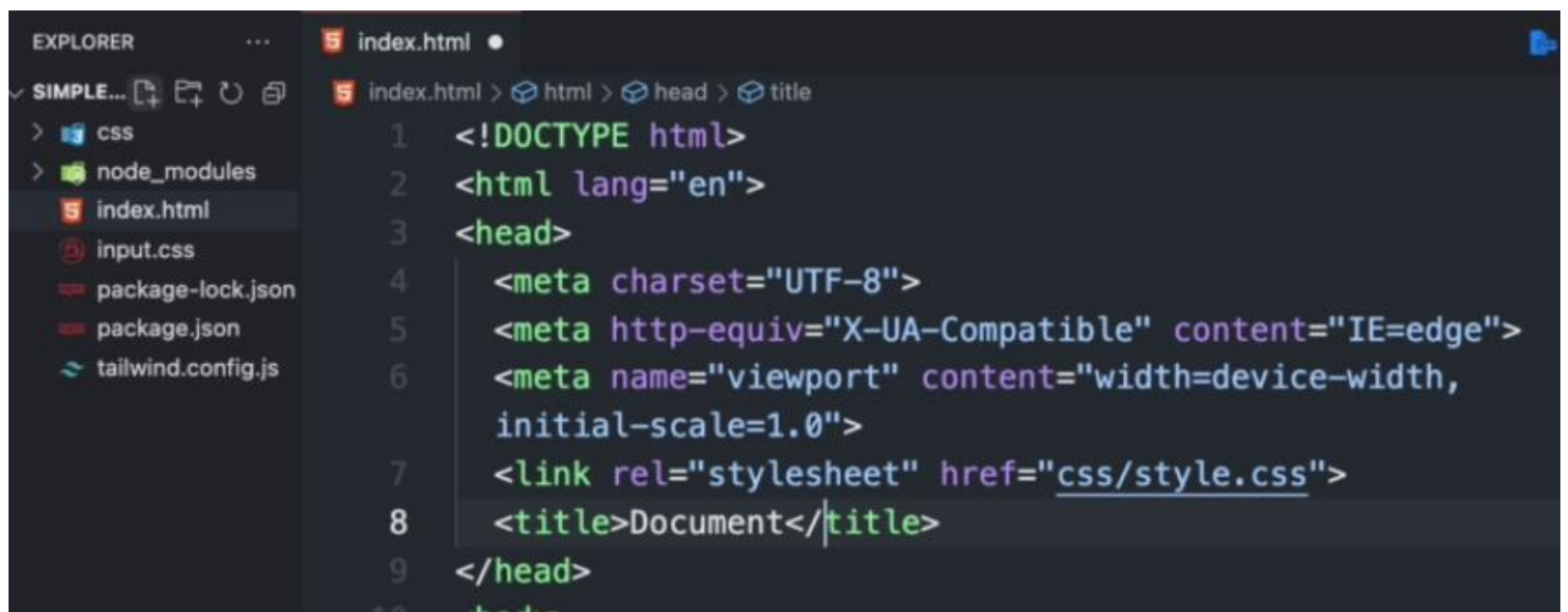
```
tailwind.config.js
1 module.exports = {
2   content: ['./*.html'],
3   theme: {
4     extend: {},
5   },
6   plugins: [],
7 }
```



```
input.css
1 @tailwind base;
2 @tailwind components;
3 @tailwind utilities;
```



```
"scripts": {
  "build": "tailwindcss -i ./input.css -o ./css/style.css",
  "watch": "tailwindcss -i ./input.css -o ./css/style.css --watch"
},
```



```
index.html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width,
7     initial-scale=1.0">
8   <link rel="stylesheet" href="css/style.css">
9   <title>Document</title>
10 </head>
11 <body>
```

Practical:

Tailwind installation k d/f methods hen CDN is not recommended,

React app banaty waqt simply tailwind ki instructions follow kro .

Jab ham kisi framework ka use nak r rahy hon to ham **Tailwind CLI** k zarye project banaty hen

So create a folder **tailwind-starter** ye banak r github pe push krdo pr jab koi naya project banana hoto download krke naya project bana lo !

Folder ko vs code me open kro or terminal pe :

npm init -y

ye package.json ki file bana degi , phr ye command:

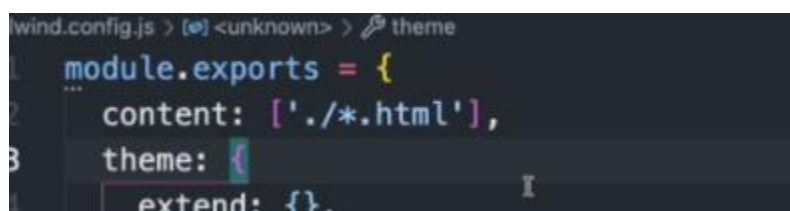
npm I -D tailwindcss

this creates a node-module folder and -lock.json file.

Then:

npx tailwindcss init

this create tailwind.config.js file where we can add extra colors and screen etc.



Root me tamam html file me tailwind ki class ko watch kro or apply krdo.

B) Now create a inputcss file on the root:

And add directives this brings main part of tailwind in our project !



Now we will create npm scripts in json file

```
"scripts": {  
  "build": "tailwindcss -i ./input.css -o ./css/style.css",  
  "watch": "tailwindcss -i ./input.css -o ./css/style.css --watch"  
},
```

Ye build cmd , input legi or output file me css code likh degi

Phr jab ham npm run build chalayb gy to root me css folder bany ga or usme style.css hogi ye tailwind css ki file ha . isko ham apni index.html me include kren gy .

Directives & Functions

We can use some things in our input.css , we can use functions and directives **@tailwind** is one example of directive but there are some others too .

Directives: Directives are custom Tailwind specific at-rules we can use in our CSS that offers special functionality for tailwind projects.

Ye main parts hen tailwind ko jo compile ho kr hamari css me jatay hen

```
@tailwind base;  
@tailwind components;  
@tailwind utilities;
```

Now in addition to those we can use **@layer** directive , we can use it for custom styles without giving classes in our html.

```
index.html  input.css  
input.css  
1  @tailwind base;  
2  @tailwind components;  
3  @tailwind utilities;  
4  
5  @layer base {  
6    h1 {  
7      font-size: 2rem;  
8    }  
9  }
```

now all of our h1 will have 2rem size.

OR we can use **@apply** directive to use existing tailwind styles

```
1 @tailwind base;
2 @tailwind components;
3 @tailwind utilities;
4
5 @layer base {
6   h1 {
7     @apply text-3xl;
8   }
9
10  h2 {
11    @apply text-xl;
12  }
13 }
```

B) Now we may want to have custom classes for higher compts like btn.

```
@layer components {
  .btn-blue {
    @apply bg-blue-500 py-2 px-4 rounded-xl font-bold text-white
    hover:bg-blue-700;
  }
}
```

```
</head>
<body>
  <h1>Simple Tailwind Starter</h1>
  <h2>Hello World</h2>
  <button class="btn-blue">Click</button>
</body>
```

We can use a couple of functions, we have themes functions and screen functions

Theme function allows us to grab config values .

```
module.exports = {
  content: ['./*.html'],
  theme: {
    extend: {
      spacing: {
        128: '32rem'
      }
    },
  },
  plugins: [],
}
```

```
<div class="content-area">
  <p>
    Lorem ipsum dolor sit amet consectetur
    adipisicing elit. Reiciendis id
    et atque molestiae alias quasi explicabo
    eligendi deserunt eos rerum
    repellat dolorem aliquam quibusdam, totam
    beatae minus pariatur,
```

Now in our input.css we can do :

```
21 .content-area {
22   @apply bg-green-200;
23   height: theme('spacing.128');
24 }
```

We are applying spacing that we defined in our config object through 'theme'...

Now we have also a screen function which allows us to specify screen breakpoints in our media query

```
26 @media screen(xl) {
27   body {
28     @apply bg-black text-white;
29   }
30 }
```
