**LESSON 7 : ADVANCED WEB DEV - INTRODUCTION TO SASS**

**Teacher Prep:**

* Go through the Document thoroughly and create the projects as guided.
* Look up the reference material links that have been provided.
* **Don't make a style.css page , SASS will generate it for you.**

**Project Showcase: (to be done at the start of the class at teacher end)**

* Open the provided project on VS Code before the class starts and check the live view on your browser.

**What is a SASS?**

* Abbreviates for Syntactically Awesome Style Sheets.
* Sass is an extension to CSS.
* Sass is a CSS preprocessor.
* Sass is completely compatible with all versions of CSS.
* Sass reduces repetition of CSS and therefore saves time.
* These files are saved with extension as **.scss**

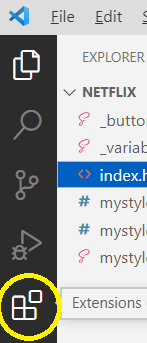
**Basic steps included:**

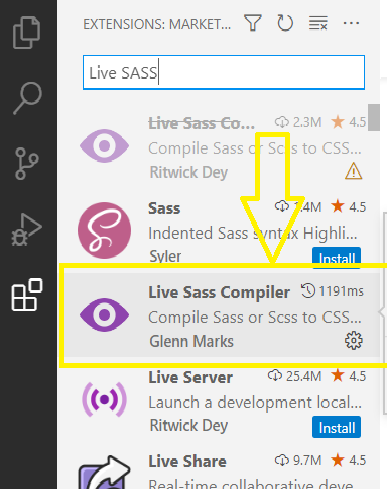
* Add Live SASS extension in your project.
* Create an HTML page.
* Create .scss pages

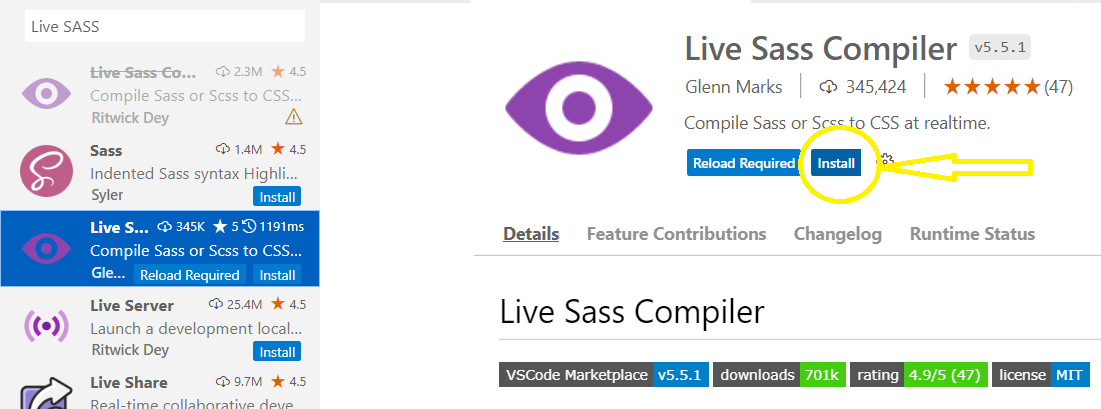
1. **Adding SASS Extension:**

Once you have made a project in VS code, goto Extensions in the left side panel of VS code.

Search for the “Live SASS” extension and install the extension.







1. **Creating HTML page:**

Create an HTML page to display the basic structure of the Netflix page.

**Initially don’t add the link tag in the <head> tag.**

Div class with “top” has the header of the page which includes an image and a Sign In button.

Next div box with the name “center” has basic headings to describe the page and a “JOIN FREE FOR A MONTH” button.

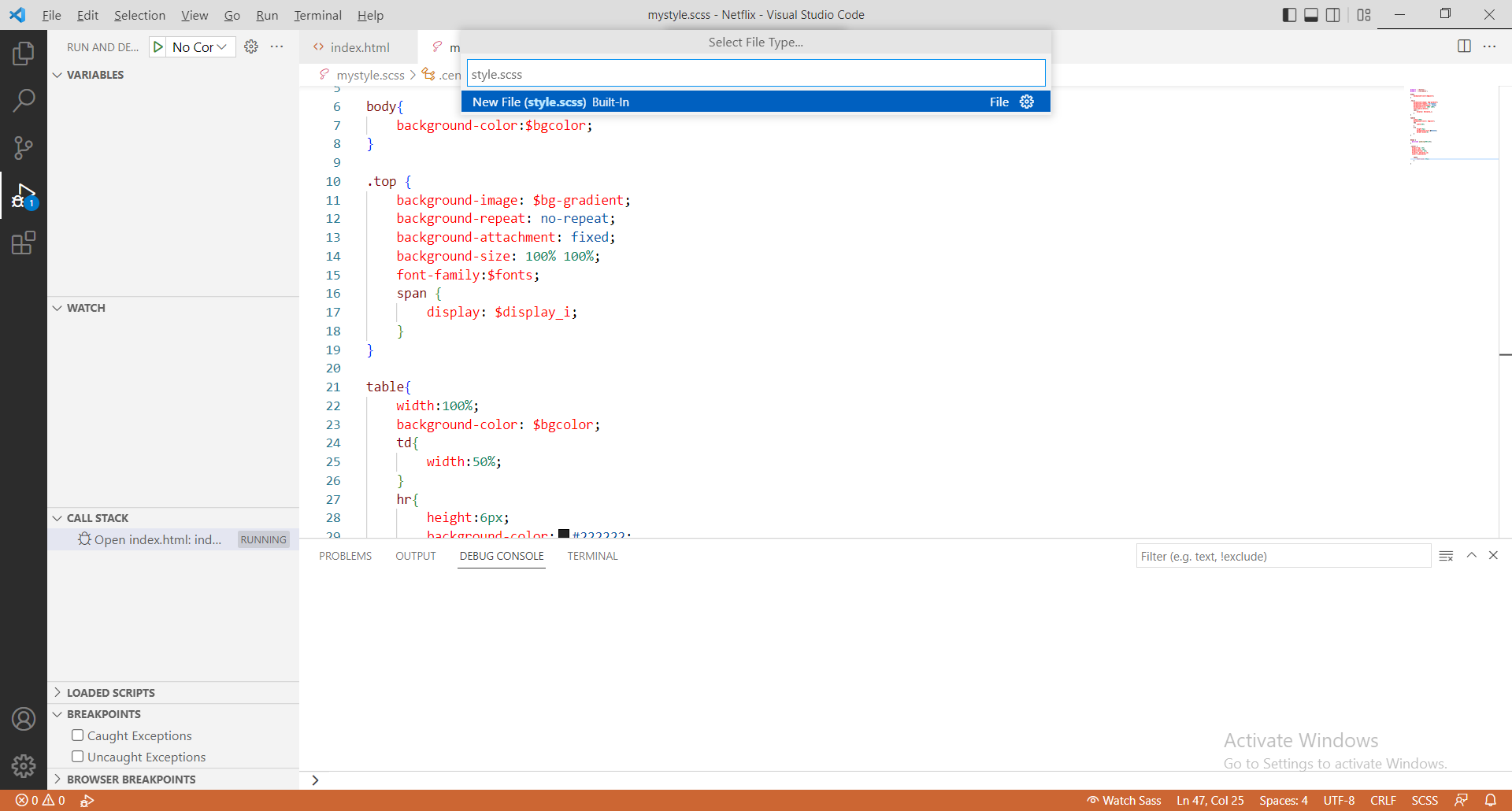
Below which , create a table for the sections.

Each <td>(Table data) in a <tr>(Table row) contains a gif image and some headings to describe it.

The description table data is given the same class name “center” as top, to give the same styling.

1. **Create a SASS file.**

Create a new file named style.scss



**Don’t directly start with @import. You can ignore it for now. Just start with variables.**

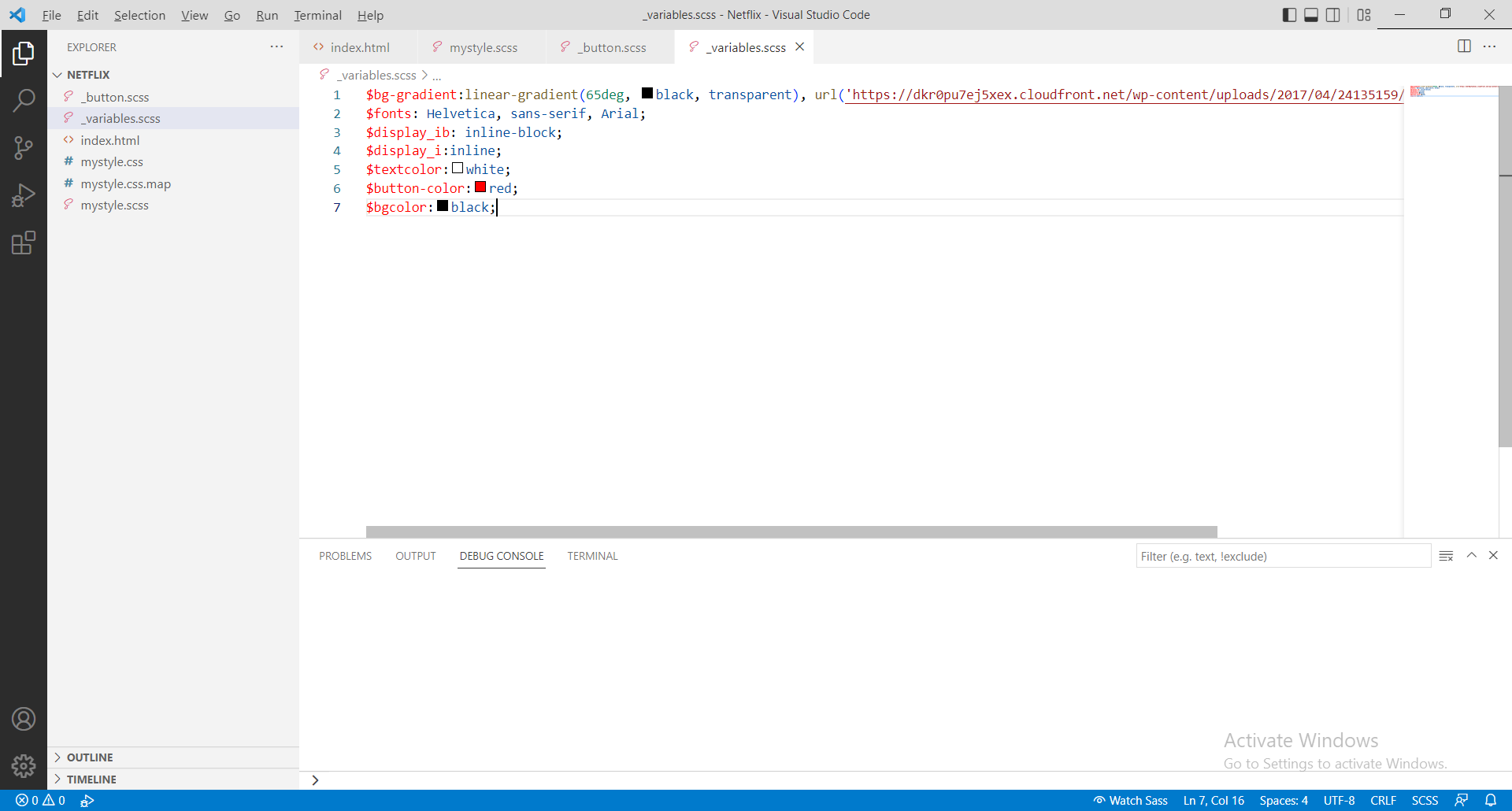
This is the SASS file where we will give the styling to our web page.

Initially you can start by adding variables in the style.css file.

The variable names has to be started with “$” symbol.

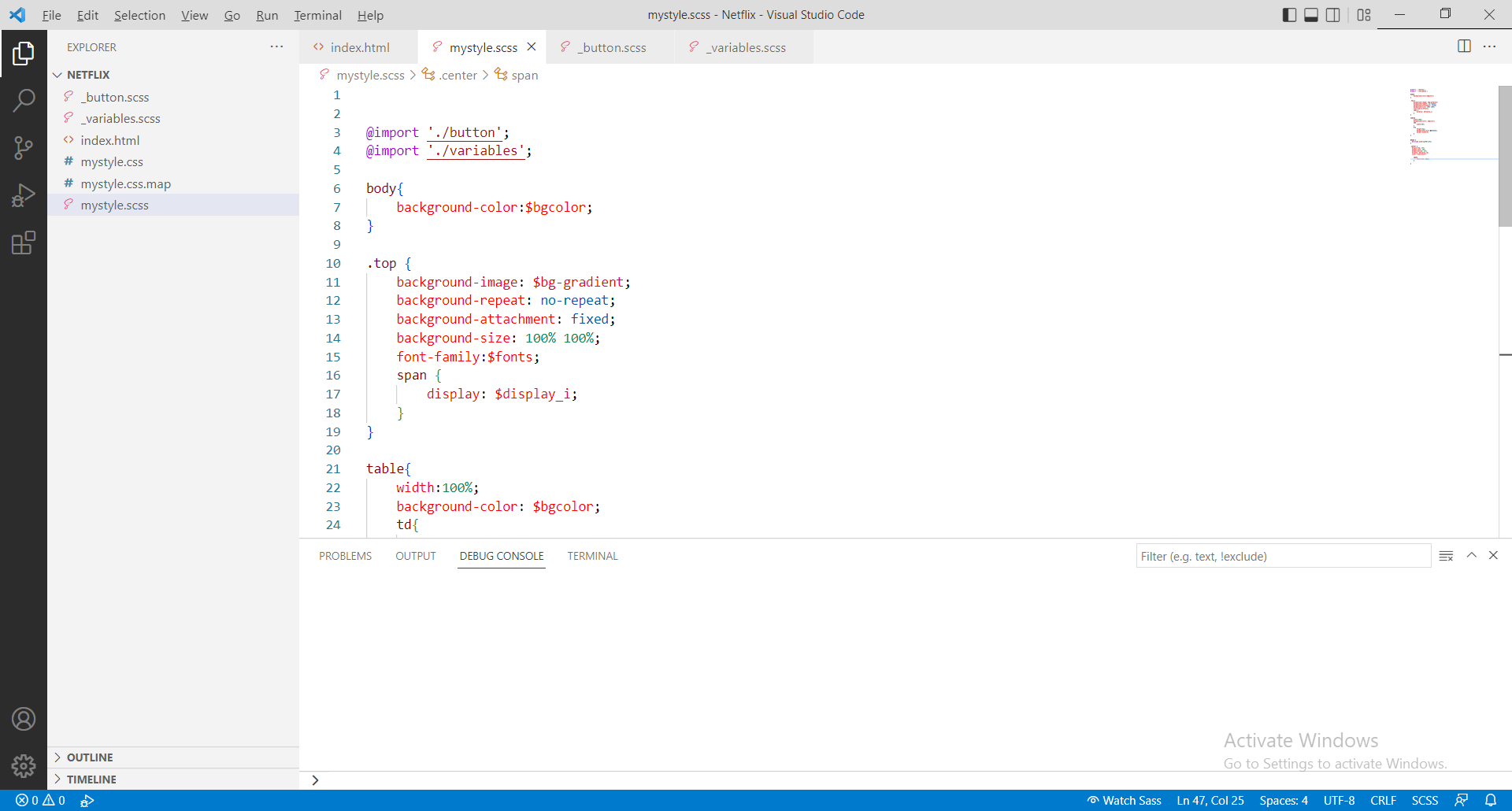
(You can copy these variables from variables.scss file and paste it in the style.scss file)

**Don’t create a separate file for now for variables.**



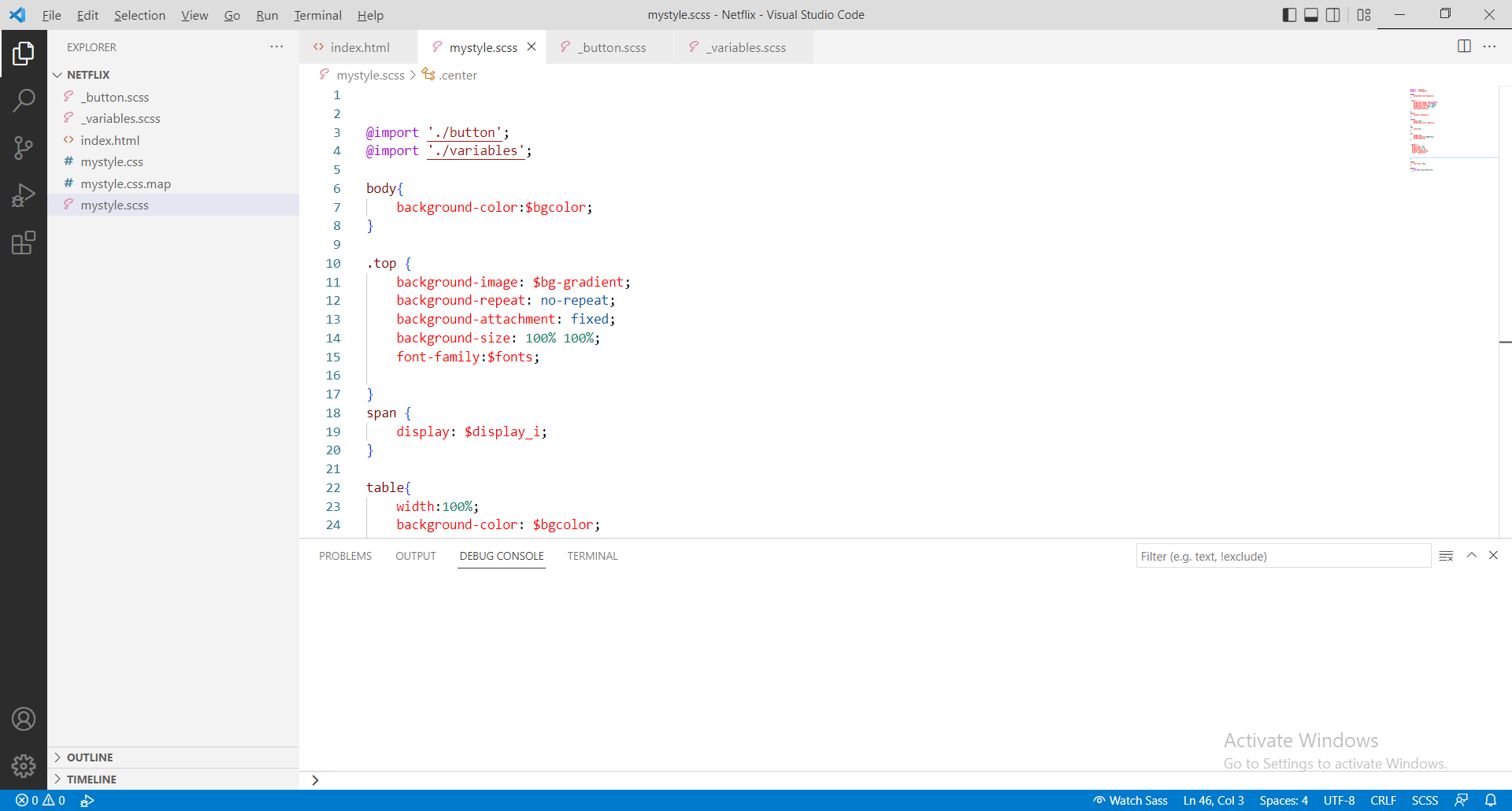
After adding all these codes in the style.scss file, you can use these variables names to give styling to our webpage.

Start with adding background color to <body> tag.



To run this:

At the bottom of the screen in VS code, you will see a button as “**Watch SASS**”.



**Once you click on this, you can see that style.css is automatically created.**

**We need not do anything on the .css file.**

**This will get updated automatically.**

And now you will have to add this line of code in your index.html page.

<head>

<link rel="stylesheet" href="mystyle.css">

</head>

Now, if you Run this, you can see the styling in your web page.

Now you can add other styling to the web page. Give styling to each tag separately and see if that works.

body{

background-color:$bgcolor;

}

.top {

background-image: $bg-gradient;

background-repeat: no-repeat;

background-attachment: fixed;

background-size: 100% 100%;

font-family:$fonts;

}

span {

display: $display\_i;

}

table{

width:100%;

background-color: $bgcolor;

}

td{

width:50%;

}

hr{

height:6px;

background-color:#222222;

border-width:0;

}

.center {

margin-top: 15%;

margin-left: 10%;

margin-bottom: 15%;

display: $display\_ib;

color: $textcolor;

}

span{

font-size: 60px;

}

**SASS NESTING:**

Nesting is **a combination of different logic structures**. Using SASS, we can combine multiple CSS rules within one another. If you are using multiple selectors, then you can use one selector inside another to create compound selectors.

In the SASS file where we have given different tags and stylings for each, we can nest them together.

Eg: As <table> tag includes <td> and <hr> tag.

We can nest them in the following way:

table{

width:100%;

background-color: $bgcolor;

td{

width:50%;

}

hr{

height:6px;

background-color:#222222;

border-width:0;

}

}

Similarly we can do for rest of the tags and classes.

body{

background-color:$bgcolor;

}

.top {

background-image: $bg-gradient;

background-repeat: no-repeat;

background-attachment: fixed;

background-size: 100% 100%;

font-family:$fonts;

span {

display: $display\_i;

}

}

table{

width:100%;

background-color: $bgcolor;

td{

width:50%;

}

hr{

height:6px;

background-color:#222222;

border-width:0;

}

}

**@import:**

What is the use of the @import function in Sass?

* The SASS @import function **helps us to import multiple SASS or CSS stylesheets together such that they can be used together**. Importing a SASS file using the @import rule allows access to mixins, variables, and functions to the other file in which the other file is imported.

We can create different .scss files for separate tags and classes.

For Example: I have created a separate Sass file for Button styling and one for storing all the variables.

**Note: While you create a new file name for each tags or classes , Give the file name as “\_filename.scss”. Ignore @mixin and @include for now.**

.button {

padding:15px,32px;

background-color: $button-color;

border: none;

color: $textcolor;

text-decoration: none;

display: $display\_ib;

font-size: 14px;

border-radius: 5px;

&:hover{

background-color:rgba(251, 122, 2, 0.912);

}

}

**Now that we have different files, we have to link them.**

**Hence to link these separate files, in the style. SCSS file, we have to use:**

@import './button';

@import './variables';

Where ‘./button’ and ‘./variables’ are the 2 different .scss files.

**@mixin**

Mixins **allow you to define styles that can be re-used throughout your stylesheet**. They make it easy to avoid using non-semantic classes like . float-left , and to distribute collections of styles in libraries.

Its similar to functions that we create in coding.

Here I have created a @mixin function with 2 parameters in the button.scss file.

@mixin padding($topbottom,$leftright)

{

padding:$topbottom,$leftright;

}

Now we can use this function all across the project wherever needed using @include

@include padding(15px,32px);

**Complete button.scss file:**

@mixin padding($topbottom,$leftright)

{

padding:$topbottom,$leftright;

}

.button {

@include padding(15px,32px);

background-color: $button-color;

border: none;

color: $textcolor;

text-decoration: none;

display: $display\_ib;

font-size: 14px;

border-radius: 5px;

&:hover{

background-color:rgba(251, 122, 2, 0.912);

}

}

.b1 {

@include padding(10px,20px )

}

**HOMEWORK:** Create a McDonalds/KFC home Page using SASS.