### SCSS

#### Objectives and Outcomes

In this exercise you will learn to write Scss code and then automatically transform it into the corresponding CSS code. At the end of this exercise, you will be able to:

* Write Scss code using many of the features of Scss
* Automatically convert the Scss code into CSS

#### Create a new React App

npx create-react-app scss-less

#### Installing and using the node-sass module

* Now we install the node module to support the compilation of the Scss file to a CSS file. To do this, type the following at the command prompt:

     npm install sass --save-dev

This will install the sass NPM module into your project and also add it as a development dependency in your package.json file.

 "devDependencies": {

    "sass": "^1.55.0"

  }

#### Rename the App.css to App.scss and import App.scss to App.js file

import './App.scss';

#### Create the Cards component

* Create the Cards.js file, update the codes as follow:

URL of images:

let img1='https://hcmuni.fpt.edu.vn/Data/Sites/1/media/2020-kim-vi/seo/campus/1-truong-dai-hoc-fpt-tphcm/truong-dai-hoc-fpt-tp-hcm-(1).jpg'

let img2 ='https://hcmuni.fpt.edu.vn/Data/Sites/1/media/2020-kim-vi/seo/campus/1-truong-dai-hoc-fpt-tphcm/truong-dai-hoc-fpt-tp-hcm-(7).jpg'

Create two cards and add some information:

#### Text Description automatically generated

#### Create SCSS folder and contain SCSS files

* A screenshot of a phone

  Description automatically generated with medium confidenceAt the **src** folder, create new folder named SCSS, in SCSS folder creates three files as follow:

\_variables.scss: to add the variables

\_mixins.scss: to add the functions that we will reuse

\_card.scss: to add the styles of our cards

#### Adding Scss Variables in \_variables.scss file

Create variables in the \_variables.scss file:

$fonts: 'Poppins', Helvetica, sans-serif;

$primary-color: #ab99ca;

$spacing: 2rem;

$dark-grey: #999;

#### Adding reusable functions in \_mixins.scss file

* For the \_mixins.scss file, this is where we'll create the reusable functions with mixins.
* Add this code to the \_mixins.scss file:

@mixin flex-center {

display: flex;

align-items: center;

justify-content: center;

}

/\* $radius is the parameter of the function \*/

@mixin border-radius($radius) {

-webkit-border-radius: $radius;

-moz-border-radius: $radius;

border-radius: $radius;

}

#### For the \_card.scss file, add this code to it:

@import '/mixins'; //import for using the border-radius in \_mixins.scss

@import '/variables'; //import for using $spacing in \_variables.scss

.card {

background-color: white;

width: 100%;

overflow: hidden;

margin: 2rem;

box-shadow: 5px 5px 5px 5px #000;

@include border-radius(0.5rem); /\* using the mixins function \*/

img {

width: 100%;

background-size: cover;

background-position: center center;

}

.card\_content {

padding: $spacing;

}

.card\_title {

margin: 0;

color: black;

}

.card\_description {

margin: 0;

color: $dark-grey;

}

&\_dark {

background-color: black;

.card\_title {

color: white;

}

}

}

In the code above we're using nesting and aliases. Nesting helps us simplify the way we write our CSS styles and allows us to nest CSS selectors.

For aliases you can use (&) or (and) followed by the class name that will resume the parent selector's code.

To use an alias, you must import it by typing the alias followed by the name of the variable (&\_dark).

#### import all the files scss into the App.scss file

* Open App.scss file and update some code as follows:

@import './scss/card';

@import './scss/mixins';

@import './scss/variables';

#root{

width: 100%;

}

.container{

width: 70%;

margin: auto;

justify-content: center;

}

body {

font-family: $fonts; /\* variable usage \*/

background-color: $primary-color;

padding: $spacing;

min-height: 100vh;

@include flex-center();

}

#### Run your React app and see the result

#### Conclusions

In this exercise you learnt to write Scss code in React App and then automatically generating the CSS file by compiling the Scss code.