



Summer Fellowship Report

On

FOSSEE WEB DEVELOPMENT

Submitted by

Sanjeevi R

Under the guidance of

Prof.Kannan M. Moudgalya

Chemical Engineering Department
IIT Bombay

June 20, 2019

Acknowledgment

We are thankful for being a part of this organisation as a fellow and being able to contribute and gain so much from the FOSSEE. I would like to thank everyone who have been instrumental in helping us achieve the desired outcome. I am very thankful to my mentor Mr. Prasanth Sinalkar for the valuable suggestions and to the Prof. Kannan M. Moudgalya, Dept of Chemical Engineering for providing us the opportunity to do a intenship wiithin the organization and to my fellow colleagues with whom i have completed the fellowship. We experienced great things together .

I perceive as this opportunity as a big milestone in my career development. I will strive to use gained skills and knowledge in the best possible way, and I will continue to work on their improvement, in order to attain desired career objectives. Hope to continue cooperation with all of you in the future.

Sincerely,

Sanjeevi.R

Contents

1 Introduction

1.1	Drupal 8	3
1.2	Task assigned	4
1.3	What's new in Drupal 8	4

2 Custom theme

2.1	File Statement	5
2.2	The .info.yml file	6
2.3	Libraries	8
2.4	Creating stylesheet	10
2.5	Adding region	12
2.6	Templates	14

3 Bootstrap

3.1	Base theme.	15
-----	---------------------	----

4 Custom thme setting

4.1	Social media	18
4.2	Flexslider	23

5 Color module

5.1 Setting up our theme.	26
5.2 Setting up our CSS.	30

Chapter 1

Introduction

1.1 Drupal 8

Drupal performs all the standard functions of the web-based content management system :

- Visitors can view published information on the site, navigate through menu, view listings, and individual pages and so on
- Users can create accounts and leave comments
- Administrators can manage the site configuration and control the permissions levels of users
- Editors can create, preview and then publish content when it is ready
- With several built-in themes, even the look and feel of the site can be change easily

With drupal 8, the scope of what a site builder can do has greatly increased.

1.2 Task Assigned

- 1) First task is about migration of FOSSEE Scilab-Arduino website from Drupal 7 to Drupal 8 and adding a custom theme to that website.
- 2) Second task is to develop the custom theme for FOSSEE projects in drupal 8.

1.3 What's New In Drupal 8

- Twig , a template engine by SensioLabs
- Classy, a new base theme
- Template.php becomes theme-name.theme
- Responsive design elements are included by default
- Breakpoints can be set and used across modules and themes

Looks at all pretty thing we get to use now!!

- HTML5
- CSS3
- Modern jQuery libraries
- SMACSS
- Standardized breakpoints

Chapter 2

Custom Theme

In simple terms, a theme is the presentational layer. Regardless of the content management system (CMS), without a theme, all you have is content that looks very much like a Word document. A theme generally consists of HTMLmarkup, CSS, JavaScript, and media (images, video, and audio).

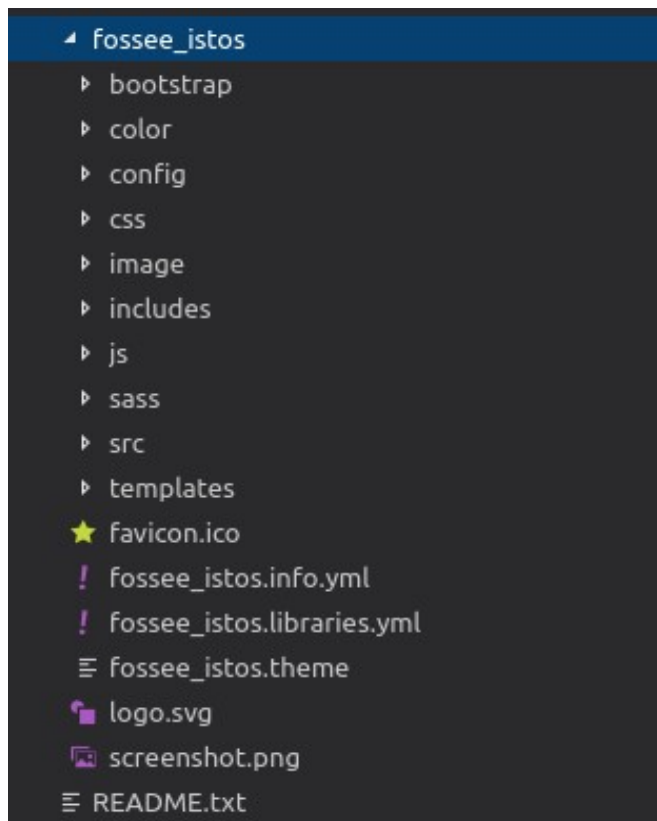
2.1 File Structure

We used to place all of themes, modules, and third-party library assets like Font-Awesome, jquery,... at the sites/all/themes directory in Drupal 7. So in the past, if you want to create a custom theme, you would place it in /sites/all/themes/{custom/}.

File structure in Drupal 8 has changed. Now, the core folder contains all the modules and themes that are used in Drupal core, and other custom or contributed modules and themes will live in the /modules, and /themes respectively.

To create a custom theme,you will need to place it at /themes/{custom}

Name your folder themename, all lowercase



File structure

This is the file structure which I have made. All the custom files and folder comes under the theme folder.

2.2 The .info.yml.file

Drupal will scan the theme directory and search for the theme_name.info.yml file to install your theme. Drupal 8 will look at the .info.yml the same way Drupal 7 looks at .info file. D8 has adopted the Symfony YAML (.yml) format.

Create a file named themename.info.yml, inside 'themename' folder.

fossee_istos.info.yml file

```
core: 8.x
type: theme
base theme: bootstrap
name: 'fossee_istos'
description: 'A Drupal Bootstrap 3 based sub-theme.'
package: 'Bootstrap'

regions:
  navigation: 'Navigation'
  navigation_collapsible: 'Navigation (Collapsible)'
  header: 'Top Bar'
  highlighted: 'Highlighted'
  help: 'Help'
  content: 'Content'
  sidebar_first: 'Primary'
  sidebar_second: 'Secondary'
  footer: 'Footer'
  footer1: 'Footer1'
  page_top: 'Page top'
  page_bottom: 'Page bottom'
  banner: 'Banner'

libraries:
  - 'fossee_istos/global-styling'
  - 'fossee_istos/color.preview'
  - 'fossee_istos/bootstrap'
  - 'fossee_istos/flexslider'
  - 'fossee_istos/fontawesome'
  - 'fossee_istos/bootstrap-scripts'
```

This is a complete info.yml file of fossee_istos theme. They will provide meta-data about your theme, and its basic functionality.

A region is basically a section on the page of your theme. You can define as many regions as you wish on your .info.yml file. And the next step you have to update your page.twig file to inform the new regions

2.3 Libraries

In Drupal 7, you would include all of your stylesheets & scripts of your theme in the .info file. Now, in Drupal 8 you have to include in both the theme_name.info.yml & theme_name.libraries.yml to enable the stylesheets, and scripts. Here are what they look like.

fossee_istos.libraries.yml file

```
global-styling:
  css:
    theme:
      css/style.css: {}
      css/colors.css: {}
  js:
    js/custom.js: {}
    js/superfish.js: {}
  dependencies:
    - core/jquery
    - core/drupal.ajax
    - core/drupal
    - core/drupalSettings
    - core/jquery.once

color.preview:
  version: VERSION
  css:
    theme:
      color/preview.css: {}
  js:
    color/preview.js: {}
  dependencies:
    - color/drupal.color

# flexslider
flexslider:
  version: 1.x
  css:
    theme:
      includes/flexslider/css/flexslider.css: {}
  js:
    includes/flexslider/js/jquery.flexslider-min.js: {}
```

fossee_istos.info.yml

```
libraries:  
  - 'fossee_istos/global-styling'  
  - 'fossee_istos/color.preview'  
  - 'fossee_istos/bootstrap'  
  - 'fossee_istos/flexslider'  
  - 'fossee_istos/fontawesome'  
  - 'fossee_istos/bootstrap-scripts'
```

In Drupal 8, if you define a library in the `.libraries.yml` file, you have to declare it in the `.info.yml` file respectively. Drupal 8 takes this approach to create the new library file in order to improve website performance. Rather than loading all CSS, JS and other assets, only those that are specified in the library are loaded.

With the example, in the `.info` file, we define a library called `global-styling`. Global-styling means that this library will be included on every page. And in the library file, we indicate the `css`, and `js` file that will load with the `global-styling` library.

Dependencies

Libraries have the ability to choose other libraries as dependencies. This is to help Drupal know what is necessary to load.

You notice that we have dependencies: core/jquery. By default, Drupal 8 does not load any scripts. JQuery is not included sidewise like in Drupal 7. So we have to inform to include the JQuery version of Drupal core. And we also define core/drupal dependencies to take advantage of Drupal behaviors

2.4 Creating Stylesheets

Add the CSS and Js files defined in themename.libraries.yml. Here, you can see an example style.css. You can design it in your own unique way.

The code simply sets the content background color, width, and margin of the navigation bar and so on. It tells about styling the text of sidebars, footer, navigation bar. You can give a unique touch to different elements of content through a stylesheet. You can also add more stylesheets as per your needs

```

/*flexslider*/
.flexslider {
  border: 0;
  max-width: 100%;
  margin: 0;
  overflow: hidden;
  box-shadow: none;
  box-shadow: none;
  box-shadow: none;
  -moz-border-radius: 0px;
  -webkit-border-radius: 0px;
  border-radius: 0px
}
.flexslider .slider-caption {
  position: absolute;
  top: 60%;
  left: 5%;
  z-index: 99;
  width: 90%;
  text-align: center;
  font-size: 1.2em;
  line-height: 150%;
  color: ■ #fff;
  padding: 20px;
  background: □ rgba(0, 0, 0, 0.3);
  border-radius: 0px;
  height: 120px;
  overflow: hidden;
}
/* social media */
body{
  padding: 0;
  margin: 0;
}
.social_media{
  color: ■ #3498db !important;
  position: absolute;
  top: 50%;
  transform: translateY(-50%);
  width: 100%;
  text-align: right;
  margin-left: -66px;
}

```

style.css

2.5 Adding regions

A region is basically a section on the page of your theme. You can define as many regions as you wish on your `.info.yml` file. And the next step you have to update your `page.html.twig` file to inform the new regions.

In order for regions to display any content placed into them, you'll need to make sure your new regions are also added to your `page.html.twig` file. Regions will be represented as Twig variables whose name corresponds with the key used in your `THEMENAME.info.yml` file with the string `page.` prepended.

Theme developers don't want to be limited with the default regions of Drupal 8. So defining regions is an inevitable task for any themers.

The first step is to declare the regions you want to add in the `.info.yml` file

The next step is to copy the `page.html.twig` file from the core templates folder and place it in a folder named `templates` within your theme (if you don't have this folder you have to create one)

```

{# Main #}
{% block main %}
<div role="main" class="main-container {{ container }} js-quickedit-main-content">
  <div class="row">

    {# Header #}
    {% if page.header %}
      {% block header %}
        <div class="col-sm-12" role="heading">
          {{ page.header }}
        </div>
      {% endblock %}
    {% endif %}

    {# Sidebar First #}
    {% if page.sidebar_first %}
      {% block sidebar_first %}
        <aside class="col-sm-3" role="complementary">
          {{ page.sidebar_first }}
        </aside>
      {% endblock %}
    {% endif %}

    {# Content #}
    {%
      set content_classes = [
        page.sidebar_first and page.sidebar_second ? 'col-sm-6',
        page.sidebar_first and page.sidebar_second is empty ? 'col-sm-9',
        page.sidebar_second and page.sidebar_first is empty ? 'col-sm-9',
        page.sidebar_first is empty and page.sidebar_second is empty ? 'col-sm-12'
      ]
    %}
  </div>

```

page.html.twig

2.6 Templates

Create a templates folder and inside which we will place our all html.twig files. Drupal allows you to override all of the templates that are used to produce HTML markup so that you can fully control the markup that is shown as output within a custom theme. There are templates for each page element ranging from the high level HTML to small fields.

Overriding templates

We can override Drupal core templates by adding templates to your theme folder that follow a specific naming convention.

To override templates you need to:

- Locate the template you wish to override.
- Copy the template file from its base location into your theme folder.
- (optionally) Rename the template according to the naming conventions in order to target a more specific subset of areas where the template is used.
- Modify the template to your liking

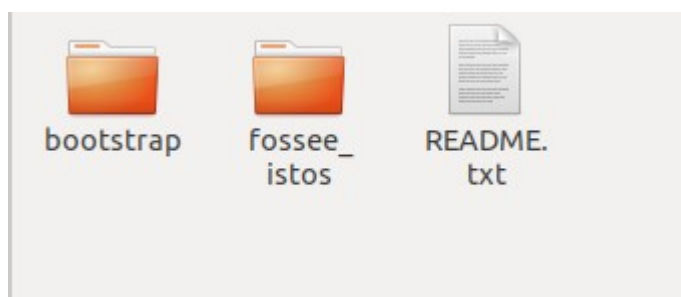
Chapter 3

Bootstrap

Bootstrap is a true blessing for web developers which is a sleek, intuitive and powerful mobile first front-end framework for faster and easier web development. When you mix that with LESS pre-processor you get a mighty tool for creating a Drupal 8 theme

3.1 Base theme

- Download, extract and place the Bootstrap base theme in your “theme” folder. It doesn’t make any difference if the theme stays disabled as we will be using it just as a parent theme for our sub-theme.



- Copy the entire folder from “/themes/bootstrap/starterkits” and place it in “/themes” along with bootstrap directory.
- Download Bootstrap Source and upload it to “fossee_istos” folder. The source directory is named as bootstrap, which contains the Source Less, JavaScript, and font files.

- Rename the following files in your “fossee_istos” theme directory:
 - THEMENAME.libraries.yml to fossee_istos.libraries.yml (All the libraries associated with your theme will be entered in this file)
 - THEMENAME.starterkit.yml to fossee_istos.info.yml (The info file for your theme)
 - THEMENAME.theme to fossee_istos.theme (Similar to template.php in Drupal 7)
 - config/install/THEMENAME.settings.yml to config/install/fossee_istos.settings.yml (This file is only used to override existing settings.)
 - config/schema/THEMENAME.schema.yml to config/schema/fossee_istos.schema.yml (Schema for the theme setting configuration file of your theme.)
- Now we need to open up a few files and perform a find and replace on the string THEMENAME.
 - Open the following files:
 - THEMENAME.info.yml: Give your sub-theme a name such as “Bootstrap” and find all THEMENAME and replace them with fossee_istos.
 - /config/schema/THEMENAME.schema.yml: Find all instances of THEMENAME and replace fossee_isto.
 - /config/install/THEMENAME.settings.yml: Find all instances of THEMENAME and replace fossee_isto.

- THEMENAME.libraries.yml: Open the libraries file and uncomment the JS section for SASS

```
bootstrap-scripts:  
  js:  
    bootstrap/js/affix.js: {}  
    bootstrap/js/alert.js: {}  
    bootstrap/js/button.js: {}  
    bootstrap/js/carousel.js: {}  
    bootstrap/js/collapse.js: {}  
    bootstrap/js/dropdown.js: {}  
    bootstrap/js/modal.js: {}  
    bootstrap/js/tooltip.js: {}  
    bootstrap/js/popover.js: {}  
    bootstrap/js/scrollspy.js: {}  
    bootstrap/js/tab.js: {}  
    bootstrap/js/transition.js: {}
```

- Then enable your theme in drupal site

Chapter 4

Custom Theme Setting

In the Drupal administration section, each theme has its own settings page at `admin/appearance/settings/themeName`. And this page has a form with standard settings like “Logo image settings” and “Shortcut icon settings.”

In Drupal 8, themes can modify the entire theme settings form by adding a PHP function to either the `THEMENAME.theme` file or to a `theme-settings.php` file. In one of those files, a theme should use `THEMENAME_form_system_theme_settings_alter` (`&$form, $form_state`) hook function.

4.1 Social_Media

- The modules provides a configurable block that display links (icons) to your profiles on various popular networking sites.
- Social media platforms such as Facebook, Twitter, LinkedIn, Instagram are easitest ways to communicate with users.
- We create new file and call it as `fossee_istos.theme` file and add the following to it or even add this to `theme-settings.php` file

The *.theme file is a PHP file that contains theme hooks for preprocessing variables. We will create a theme file specific to our theme that we can use to grab the comment count, based on each individual post, and then return the count to our Twig template as a variable that can be printed.

fossee_istos.theme file

Implementing the hook_preprocess_page for block template

```
1  <?php
2
3  /**
4   * @file
5   * Bootstrap sub-theme.
6   *
7   * Place your custom PHP code in this file.
8   */
9
10 use Drupal\Core\Template\RenderWrapper;
11 use Drupal\Core\Template\Attribute;
12 use Drupal\search\Form\SearchBlockForm;
13 use Drupal\Component\Utility\SafeMarkup;
14 use Drupal\Core\Form\FormStateInterface;
15 use Drupal\system\Form\ThemeSettingsForm;
16 use Drupal\file\Entity\File;
17 use Drupal\Core\Url;
18 use Drupal\file\Plugin\Core\Entity\FileInterface;
19
20 /**
21  * Implements hook_preprocess_page() for block templates.
22  */
23 function fossee_istos_preprocess_block(&$variables) {
24     global $base_path, $theme_name;
25
26     // Social media global variable.
27     $variables['show_social_icon'] = theme_get_setting('show_social_icon');
28     $variables['facebook_url'] = theme_get_setting('facebook_url');
29     $variables['google_plus_url'] = theme_get_setting('google_plus_url');
30     $variables['twitter_url'] = theme_get_setting('twitter_url');
31     $variables['linkedin_url'] = theme_get_setting('linkedin_url');
32 }
```

Implementing hook_form_system_alter() function

```
/**
 * Implements hook_form_system_theme_settings_alter().
 */
function fossee_istos_form_system_theme_settings_alter(&$form, \Drupal\Core\Form\FormStateInterface $form_state) {

  //Social Icon Link
  $form['fossee_istos_settings']['social_icon'] = array(
    '#type' => 'details',
    '#title' => t('Social Media Link'),
    '#collapsible' => TRUE,
    '#collapsed' => FALSE,
  );
  $form['fossee_istos_settings']['social_icon']['show_social_icon'] = array(
    '#type' => 'checkbox',
    '#title' => t('Show Social Icons'),
    '#default_value' => theme_get_setting('show_social_icon'),
    '#description' => t("Show/Hide Social media links"),
  );
  $form['fossee_istos_settings']['social_icon']['facebook_url'] = array(
    '#type' => 'textfield',
    '#title' => t('Facebook URL'),
    '#default_value' => theme_get_setting('facebook_url'),
  );
  $form['fossee_istos_settings']['social_icon']['google_plus_url'] = array(
    '#type' => 'textfield',
    '#title' => t('Google plus URL'),
    '#default_value' => theme_get_setting('google_plus_url'),
  );
  $form['fossee_istos_settings']['social_icon']['twitter_url'] = array(
    '#type' => 'textfield',
    '#title' => t('Twitter URL'),
    '#default_value' => theme_get_setting('twitter_url'),
  );
}
```

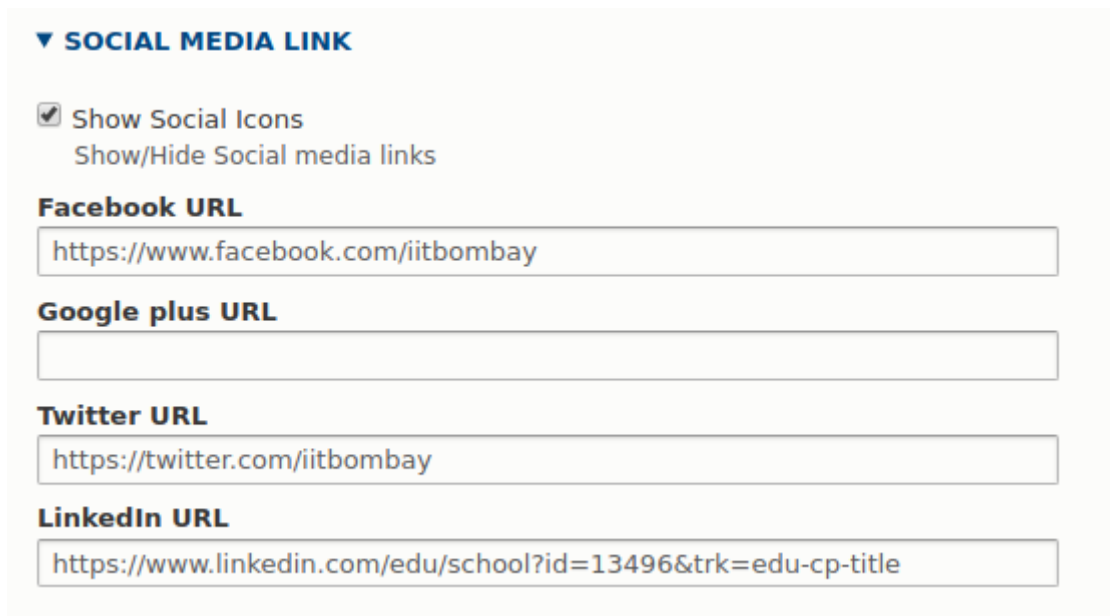
Then we want to use the variable in the .twig file

```
<div class="social_media">
  {% if show_social_icon %}
    {% if facebook_url %}
      <a class="social" href="{{ facebook_url }}" class="facebook" target="_blank" title="{{ 'Facebook'|t }}" ><i
    {% endif %}
    {% if google_plus_url %}
      <a class="social" href="{{ google_plus_url }}" class="google-plus" target="_blank" title="{{ 'Google plus'|t
    {% endif %}
    {% if twitter_url %}
      <a class="social" href="{{ twitter_url }}" class="twitter" target="_blank" title="{{ 'Twitter'|t }}"><i class
    {% endif %}
    {% if linkedin_url %}
      <a class="social" href="{{ linkedin_url }}" class="linkedin" target="_blank" title="{{ 'Linked In'|t }}"><i
    {% endif %}
  </p>
{% endif %}
</div>
```

Then i have given the default value for the social url, we can include it in our config/install/fossee_istos.settings.yml

```
#default value for social media
facebook_url: 'https://www.facebook.com/iitbombay'
twitter_url: 'https://twitter.com/iitbombay'
google_plus_url: 'https://plus.google.com/+iitbombay'
linkedin_url: 'https://www.linkedin.com/edu/school?id=13496&trk=edu-cp-title'
```

Once that file is there you can clear your site's cache and go to the theme settings for your site /admin/appearance/settings/fosee_istos and you will see the social media form



▼ SOCIAL MEDIA LINK

Show Social Icons
Show/Hide Social media links

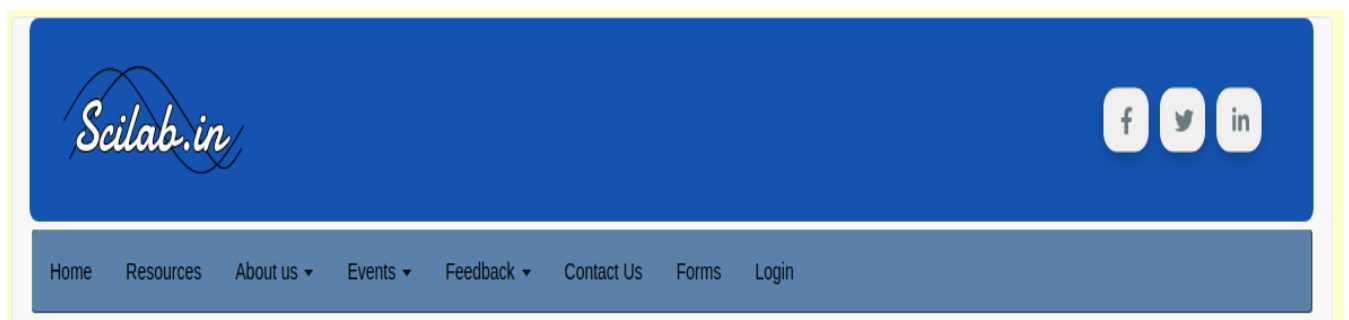
Facebook URL

Google plus URL

Twitter URL

LinkedIn URL

Site administrator can simply add the url in the text box for particular social media icons then it appear in the site header.if the admin not define the socil media url then the google-plus icon will not show in the site.



4.2 Flexslider

The module allows you to display multiple images as a slideshow within a single node. Flexslider is a popular jQuery responsive slider that provides designers and developers a fast way to get up and running with an image slider.

4.2.1 Adding Flexslider Files

- Download the files from <http://flexslider.woothemes.com/>
- Extract Flexslider to our include folder in our theme.
- Give the path to the Flexslider in the info.yml file and the .libraries.yml file
- And then implement the custom function and the hook function in the fosse_istos.theme file
- Then we want to use the variable in the .twig file

Implements custom_form_system_theme_settings_alter()

```
/**
 * Slider
 * Implements custom function for get slider content.
 */

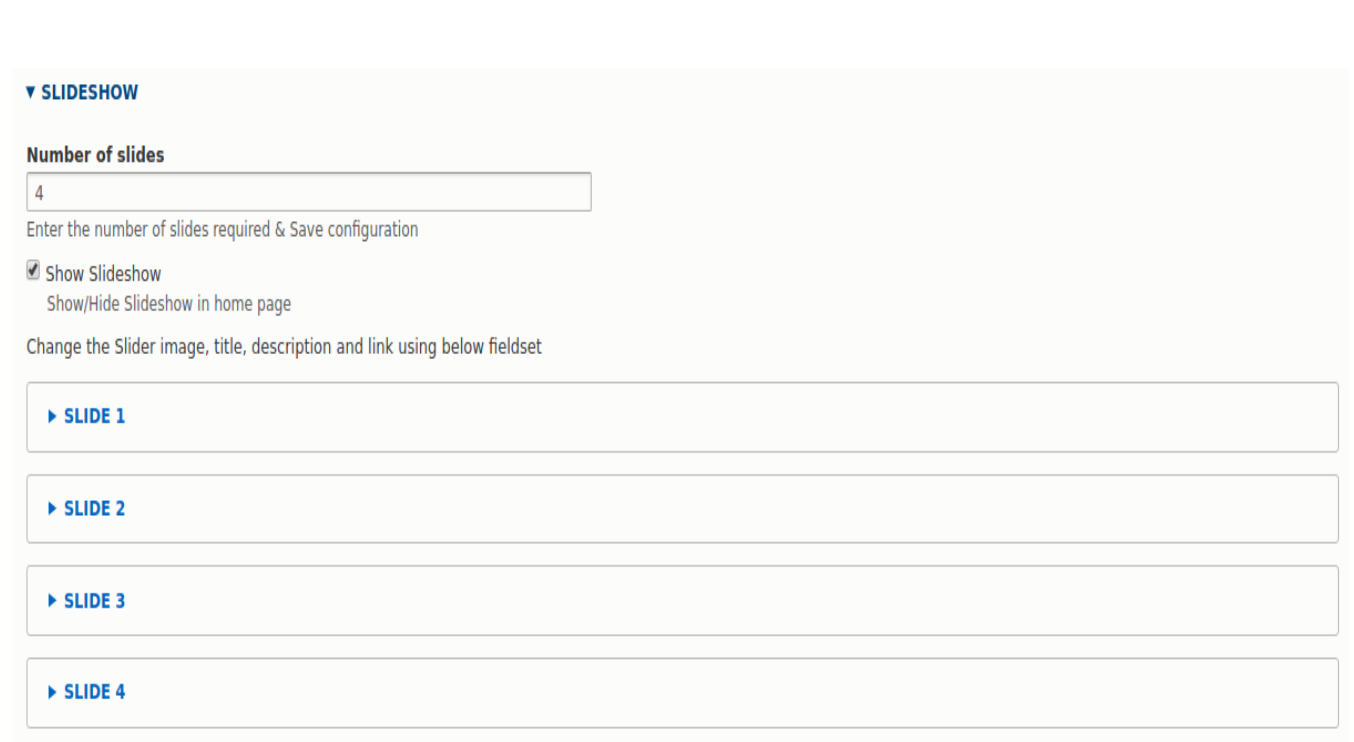
function fossee_istos_get_slider_content() {
  $slider_content = array();
  for ($i = 1; $i <= theme_get_setting('no_of_slides'); $i++) {
    $fid = theme_get_setting('slide_image_path' . $i, 'fossee_istos');
    if (!empty($fid)) {
      $file = file_load($fid[0]);
      $uri = $file->getFileUri();
      $path = file_create_url($uri);
    }
    else {
      $path = base_path() . drupal_get_path('theme', 'fossee_istos') . theme_get_setting('slide_image_
    }
    $slider_content[$i] = '<li>
      <img src="" . $path . "" alt="loading..." />
      <div class="slider-caption">
        <h2 class="slider-title">' . theme_get_setting('slide_title_' . $i, 'fossee_istos'). '</h2>
        <div class="clearfix"><p class="slider-description">' . theme_get_setting('slide_description_'
        <a href=' . theme_get_setting('slide_url_' . $i, 'fossee_istos') . ' class="more-link">Read mo
      </div>
    </li>';
  }
  return $slider_content;
}
```

Implements hook_form_system_theme_settings_alter()

```
//Slide show configure
$form['fossee_istos_settings']['slideshow'] = array(
  '#type' => 'details',
  '#title' => t('Slideshow'),
  '#collapsible' => TRUE,
  '#collapsed' => FALSE,
);
$form['fossee_istos_settings']['slideshow']['no_of_slides'] = array(
  '#type' => 'textfield',
  '#title' => t('Number of slides'),
  '#default_value' => theme_get_setting('no_of_slides'),
  '#description' => t("Enter the number of slides required & Save configuration"),
  '#markup' => '<div class="messages messages--warning">Clear caches after making any changes in theme
);
$form['fossee_istos_settings']['slideshow']['show_slideshow'] = array(
  '#type' => 'checkbox',
  '#title' => t('Show Slideshow'),
  '#default_value' => theme_get_setting('show_slideshow'),
  '#description' => t("Show/Hide Slideshow in home page"),
);
$form['fossee_istos_settings']['slideshow']['slide'] = array(
  '#markup' => t('Change the Slider image, title, description and link using below fieldset'),
);

for ($i = 1; $i <= theme_get_setting('no_of_slides'); $i++) {
  $form['fossee_istos_settings']['slideshow']['slide' . $i] = array(
    '#type' => 'details',
    '#title' => t('Slide ' . $i),
    '#collapsible' => TRUE,
    '#collapsed' => TRUE,
  );
  $form['fossee_istos_settings']['slideshow']['slide' . $i]['slide_image_path' . $i] = array(
    '#type' => 'managed_file'
  );
}
```

Once that file is there you can clear your site's cache and go to the theme settings for your site /admin/appearance/settings/fosee_istos and you will see the flexslder form



The screenshot shows a configuration form for a Flexslider. At the top, there is a section header "▼ SLIDESHOW". Below it, the "Number of slides" is set to 4 in a text input field. A note below the field says "Enter the number of slides required & Save configuration". There is a checked checkbox for "Show Slideshow" with a sub-note "Show/Hide Slideshow in home page". Below this, a text label reads "Change the Slider image, title, description and link using below fieldset". The form contains four stacked text input fields, each with a blue arrow icon and the label "SLIDE 1", "SLIDE 2", "SLIDE 3", and "SLIDE 4" respectively.

Here admin can give the flexslider image, title, caption, url then it will display in the front page. Admin can able to change it dynamically

Chapter 5

Color Module

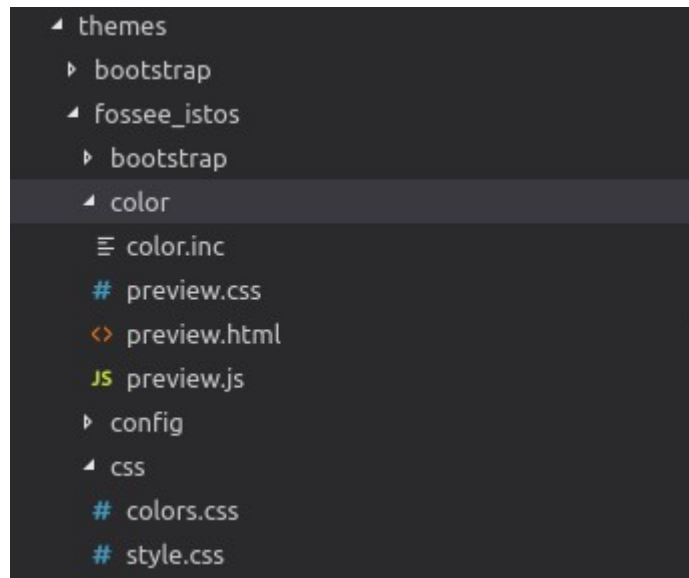
The color module built into Drupal 8 core allows administrators to change the color scheme of compatible themes. Creating a theme that need to work for multiple sites with different color schemes .The Color module allows you to easily change the color of links, backgrounds, text, and other theme elements. Now admins can choose whatever colors they want.

5.1 Setting up our Theme

Once we have created our theme we need to create the directory color and we want to create at least one file named color.inc . There are more files that can be put here to make the preview for the admins better like preview.css, preview.js, preview.html

We also need a CSS file with all of your color things that you are letting the admins control with Drupal.

Our directory structure should look like the following example. Where your CSS file is and what it is named isn't a requirement of the Color Module, but you do need to properly link to them.



The fossee_istos.libraries.yml is needed to give the path to your CSS that the Color Module will be using. It should look like the following:

```
color.preview:  
  version: VERSION  
  css:  
    theme:  
      color/preview.css: {}  
  js:  
    color/preview.js: {}  
  dependencies:  
    - color/drupal.color
```

The fossee_istos.info.yml is a requirement of every theme, and the only thing worth mentioning here is that it must have a library that it is using so you can pull in your CSS changes. It should look like the following.

```
libraries:  
  - 'fossee_istos/global-styling'  
  - 'fossee_istos/color.preview'  
  - 'fossee_istos/bootstrap'  
  - 'fossee_istos/flexslider'  
  - 'fossee_istos/fontawesome'  
  - 'fossee_istos/bootstrap-scripts'
```











The color.inc is the file that is doing all of the work, a minimum file would be like the following:

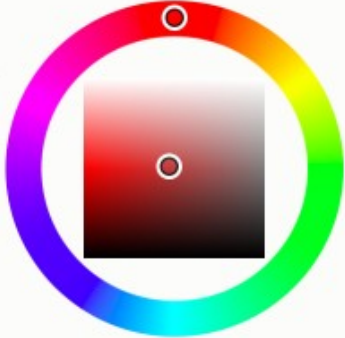
```
$info = [  
  // Available colors and color labels used in theme.  
  'fields' => [  
    'topnav' => t('Header and Footer Text color'),  
    'footer' => t('Header and Footer background'),  
    'top' => t('Navbar background'),  
    'nav-text' => t('Navbar text'),  
    'nav-hover' => t('Navbar hover'),  
    'main' => t('Main background'),  
    'bg' => t('Text background'),  
    'sidebar' => t('Sidebar background'),  
    'text' => t('Default Text color'),  
    'link' => t('Link color'),  
  ],  
  // Pre-defined color schemes.  
  'schemes' => [  
    'default' => [  
      'title' => t('Fossee (default)'),  
      'colors' => [  
        'topnav' => '#ffffff',  
        'footer' => '#444444',  
        'top' => '#3498db',  
        'nav-text' => '#000000',  
        'nav-hover' => '#e6e6e6',  
        'main' => '#fffacd',  
        'bg' => '#e0ffff',  
        'sidebar' => '#f6f6f2',  
        'text' => '#343536',  
        'link' => '#337ab7',  
      ],  
    ],  
  ],  
],
```

Once that file is there you can clear your site's cache and go to the theme settings for your site /admin/appearance/settings/fosee_istos and you will see the color form. The form should have all of the fields you defined and show a color wheel allowing admin to pick whatever color they'd like for the page

▼ COLOR SCHEME

Color set

Header and Footer Text color	<input type="text" value="#c54040"/>	
Header and Footer background	<input type="text" value="#444444"/>	
Navbar background	<input type="text" value="#3498db"/>	
Navbar text	<input type="text" value="#000000"/>	
Navbar hover	<input type="text" value="#ebe1ce"/>	
Main background	<input type="text" value="#fffacd"/>	
Text background	<input type="text" value="#e0ffff"/>	
Sidebar background	<input type="text" value="#f6f6f2"/>	
Default Text color	<input type="text" value="#343536"/>	
Link color	<input type="text" value="#337ab7"/>	



Now admins can choose whatever colors they want. If you change those and view the site, you will see that has happened yet. The next step needed is to create the CSS that Drupal will actually be using to color the site with the values entered in the admin form.

4.2 Setting up our CSS

Create color classes that you apply to your HTML as needed that color things.

```
.main-container.container.js-quickedit-main-content {
  background: #e0ffff;
}

.dialog-off-canvas-main-canvas {
  background: #fffacd ;
}

#block-views-block-custom-block-block-1-2
{
  background: #f6f6f2;
}

.region.region-sidebar-second {
  background: #f6f6f2;
}

.menu.menu--custom-menu.nav {
  background: #f6f6f2;
}

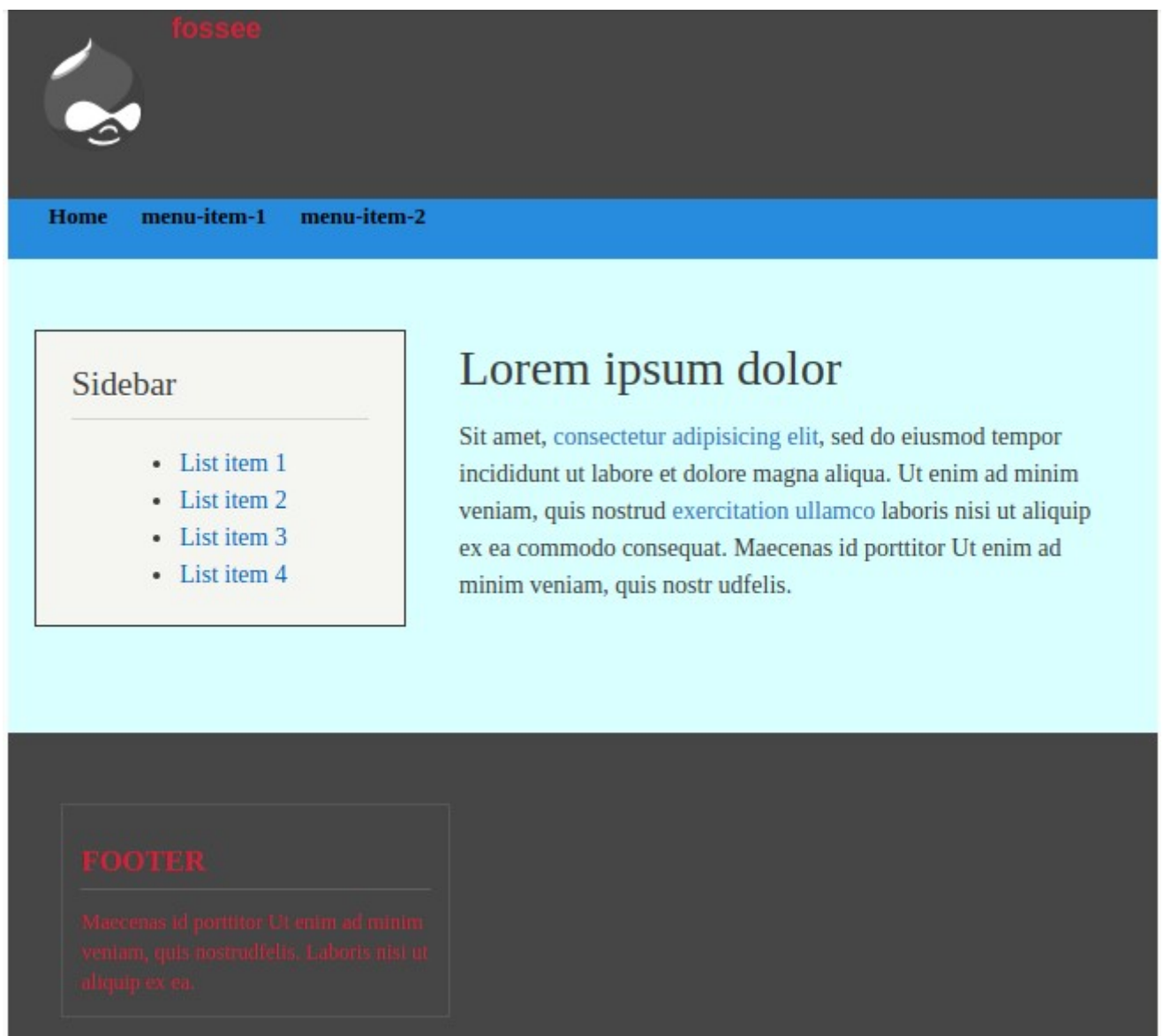
.menu--main > li > a:hover, .nav > li > a:focus {
  background-color: #e0e0e0 !important;
}

ul.menu.menu--main.nav.navbar-nav {
  background: #3498db ;
}

ul.dropdown-menu {
  background: #3498db !important;
}
```


Something very important that you should take note of is how colors in your CSS map to the colors in the admin form. In Drupal 8 we are grabbing the HEX defined as the default in your color.inc and replacing that with the new value saved in the config form. In our default scheme, .main-container is set to #e0ffff so in our color.css file anywhere that hex exists, it will get overridden to be the new primary color set in the admin interface.

Result :



Reference

- <https://stackoverflow.com/>
- <https://www.drupal.org/>
- Various resources on youtube