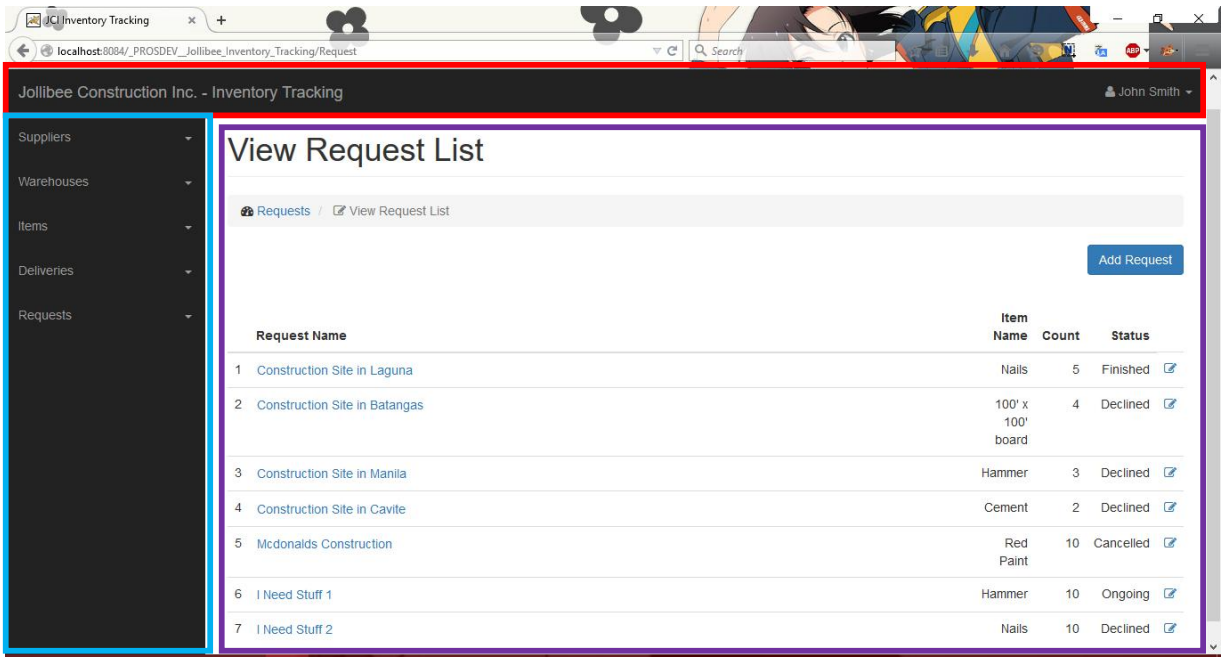
Yoooo makotos!

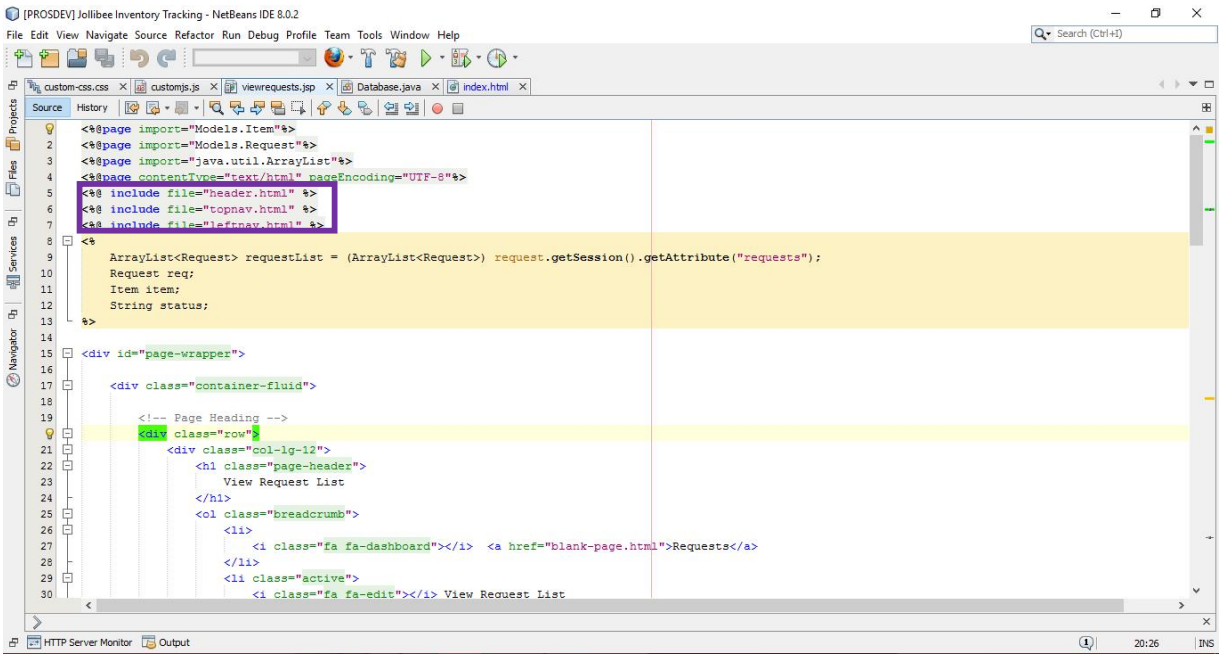
Let's learn front-end using Bootstrap Admin!

First, let's see how a usual page in Bootstrap Admin looks like. A perfect example is viewrequest.jsp:



There are two parts of the page, the navigation and the body. The navigation is split into two: the top bar and the side bar. The top bar is highlighted in red, the side bar in blue and the body in violet.

Now how is it organized? Well, notice these three page imports:



Why import these three pages? Well, it's similar to the "include" function in php, where you can insert other pages. This is to reduce copy-pasting common features in pages and so as to easily edit them. Imagine if we didn't use include file, we would have to copy-paste the contents of header.html, topnav.html and leftnav.html.

Now what do they contain? Let's start with header.html:

```
<!DOCTYPE html>

<html lang="en">

    <head>

        <meta charset="utf-8">

        <meta http-equiv="X-UA-Compatible" content="IE=edge">

        <meta name="viewport" content="width=device-width, initial-
scale=1">

        <meta name="description" content="">

        <meta name="author" content="">

        <title>JCI Inventory Tracking</title>

        <!--Custom CSS-->

        <link href="css/custom-css.css" rel="stylesheet">

        <!-- Bootstrap Core CSS -->

        <link href="css/bootstrap.min.css" rel="stylesheet">

        <!-- Custom CSS -->

        <link href="css/sb-admin.css" rel="stylesheet">

        <!-- Custom Fonts -->

        <link href="font-awesome/css/font-awesome.min.css" rel="stylesheet"
type="text/css">

        <!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and
media queries -->

        <!-- WARNING: Respond.js doesn't work if you view the page via
file:// -->

        <!--[if lt IE 9]>

            <script
src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>

            <script
src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>

        <![endif]-->

    </head>

    <body>
```

So, from what we're seeing here, header.html contains the beginning <html> and <body> tags. It also contains the header, where several css files are included as well. DO NOT delete anything you see here, every tag and reference is important. In order to add custom classes on top of Bootstrap's classes, only edit custom-css.css.

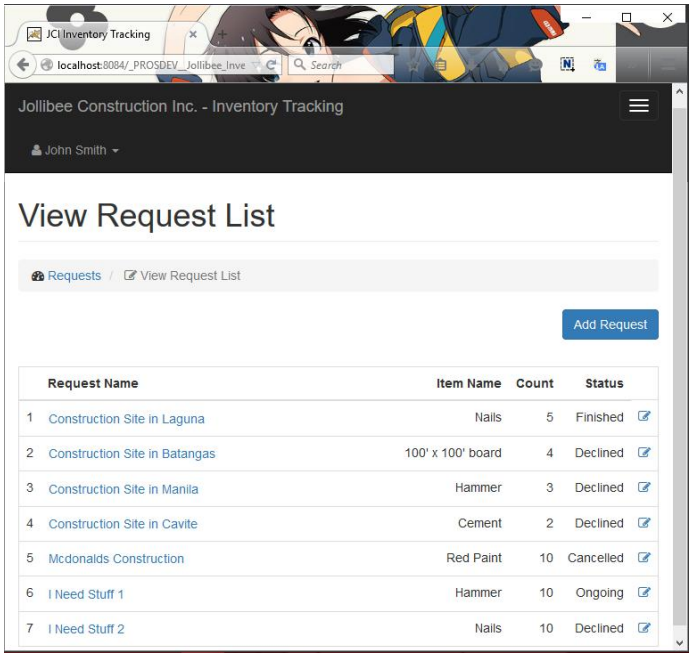
Do you still remember the three parts of the Bootstrap Admin page? Let's talk about navigation, where users can go from page to page. There are two parts to navigation: the top bar and the side bar. Let's focus on the top bar first. Let's divide it into four parts: the wrapper, top bar header and the top right navigation.

The first part is called the wrapper, which contains all the other three parts. There are two parts to this: the main wrapper and navigation header. The main wrapper is the `<div id="wrapper">` and `</div>` found in the beginning and the end of this and every other major page. The class 'wrapper' ensures the part's responsiveness towards all viewport sizes (e.g. browsers, phone screens). The navigation header is `<nav class="navbar navbar-inverse navbar-fixed-top" role="navigation">` and `</nav>`.

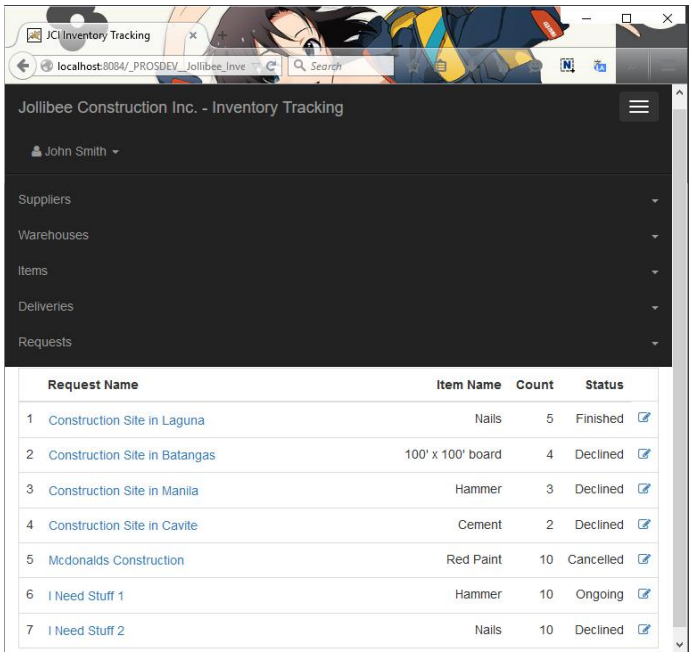
The second part is the top bar header. Its HTML is as follows:

```
<!-- Brand and toggle get grouped for better mobile display -->

      <div class="navbar-header">

            <button type="button" class="navbar-toggle" data-
toggle="collapse" data-target=".navbar-ex1-collapse">

                  <span class="sr-only">Toggle navigation</span>

                  <span class="icon-bar"></span>

                  <span class="icon-bar"></span>

                  <span class="icon-bar"></span>

             </button>

             <a class="navbar-brand" href="index.html">Jollibee
Construction Inc. - Inventory Tracking</a>

     </div>
```

Notice the screenshot I took of viewrequest.jsp. In that screenshot, the button shown here does not appear on the page because the viewport is too wide. It can only be seen when its viewport is small enough, as shown below:



When you click it, you can see the contents of the side bar pop up:



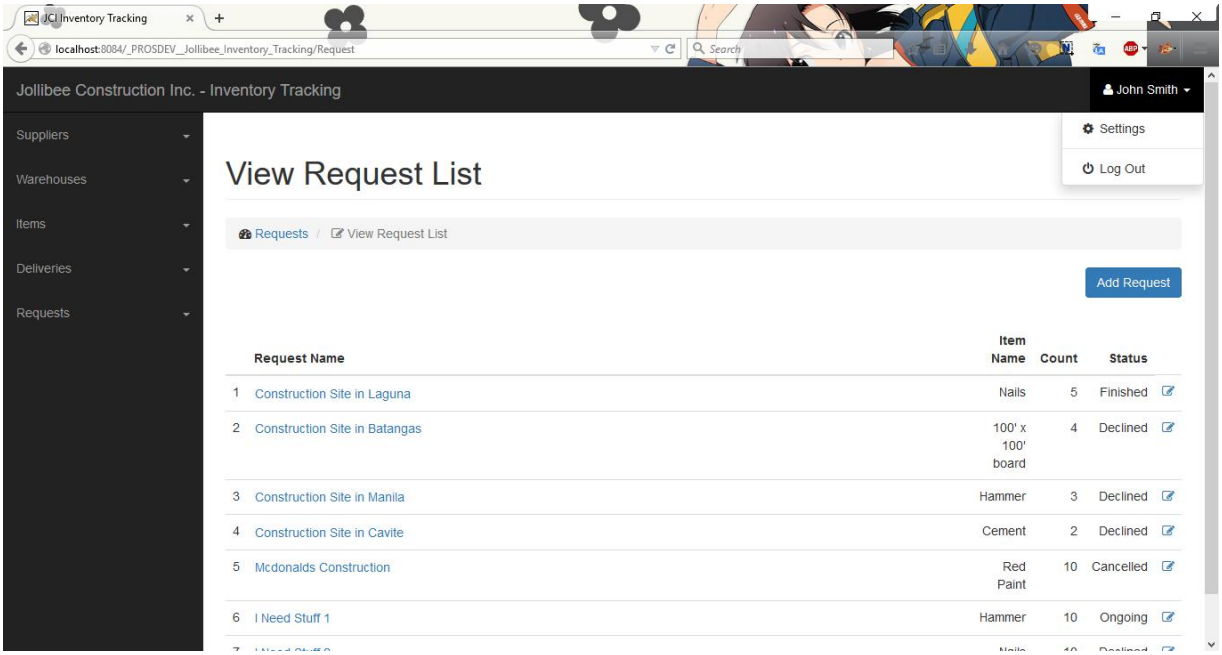
Meaning that this special HTML button will only be shown to guide users to the pages found in this admin in smaller viewports.

The <a> link right after the button is called the brand and it is basically named after the name of the system, which in this case is 'Jollibee Construction Inc. – Inventory Tracking'. When clicked, it leads to the homepage (index.html).

The third part of the top bar is called the top right navigation. Its HTML is as follows:

```
<!-- Top Menu Items -->

      <ul class="nav navbar-right top-nav">

            <li class="dropdown">

                  <a href="#" class="dropdown-toggle" data-
toggle="dropdown"><i class="fa fa-user"></i> John Smith <b
class="caret"></b></a>

                  <ul class="dropdown-menu">

                        <li>

                              <a href="#"><i class="fa fa-fw fa-gear"></i>
Settings</a>

                        </li>

                        <li class="divider"></li>

                        <li>

                        <a href="#"><i class="fa fa-fw fa-power-off"></i>
Log Out</a>

                        </li>

                  </ul>

            </li>

      </ul>
```

If it isn't obvious enough, this part is found at the top right corner of the top bar. This is where you can place user information pages such as Settings and Logout. The name and image of the person you see of the current user logged in. At the moment, 'John Smith' is the user for all pages and the 'Settings' and 'Log out' link are mere placeholders. A view of the top right navigation, with its page list shown, can be seen below:


JCI Inventory Tracking
localhost:8084/_PROSDEV__Jollibee_Inventory_Tracking/Request
Search
Jollibee Construction Inc. - Inventory Tracking
John Smith
Settings
Log Out
Suppliers
Warehouses
Items
Deliveries
Requests
View Request List
Requests / View Request List
Add Request
Request Name
Item Name
Count
Status
1 Construction Site in Laguna Nails 5 Finished
2 Construction Site in Batangas 100' x 100' board 4 Declined
3 Construction Site in Manila Hammer 3 Declined
4 Construction Site in Cavite Cement 2 Declined
5 Mcdonalds Construction Red Paint 10 Cancelled
6 I Need Stuff 1 Hammer 10 Ongoing

All together, topnav.html looks like this:

```
<div id="wrapper">

      <!-- Navigation -->

      <nav class="navbar navbar-inverse navbar-fixed-top"
role="navigation">

      <!-- Brand and toggle get grouped for better mobile display -->

            <div class="navbar-header">

                  <button type="button" class="navbar-toggle" data-
toggle="collapse" data-target=".navbar-ex1-collapse">

                        <span class="sr-only">Toggle navigation</span>

                        <span class="icon-bar"></span>

                        <span class="icon-bar"></span>

                        <span class="icon-bar"></span>

                    </button>

                    <a class="navbar-brand" href="index.html">Jollibee
Construction Inc. - Inventory Tracking</a>

                </div>

                <!-- Top Menu Items -->

                <ul class="nav navbar-right top-nav">

                    <li class="dropdown">

                        <a href="#" class="dropdown-toggle" data-
toggle="dropdown"><i class="fa fa-user"></i> John Smith <b
class="caret"></b></a>

                        <ul class="dropdown-menu">

                            <li>

                                <a href="#"><i class="fa fa-fw fa-
gear"></i> Settings</a>

                            </li>

                            <li class="divider"></li>

                            <li>

                                <a href="#"><i class="fa fa-fw fa-power-
off"></i> Log Out</a>

                            </li>

                        </ul>

                    </li>

                </ul>
```

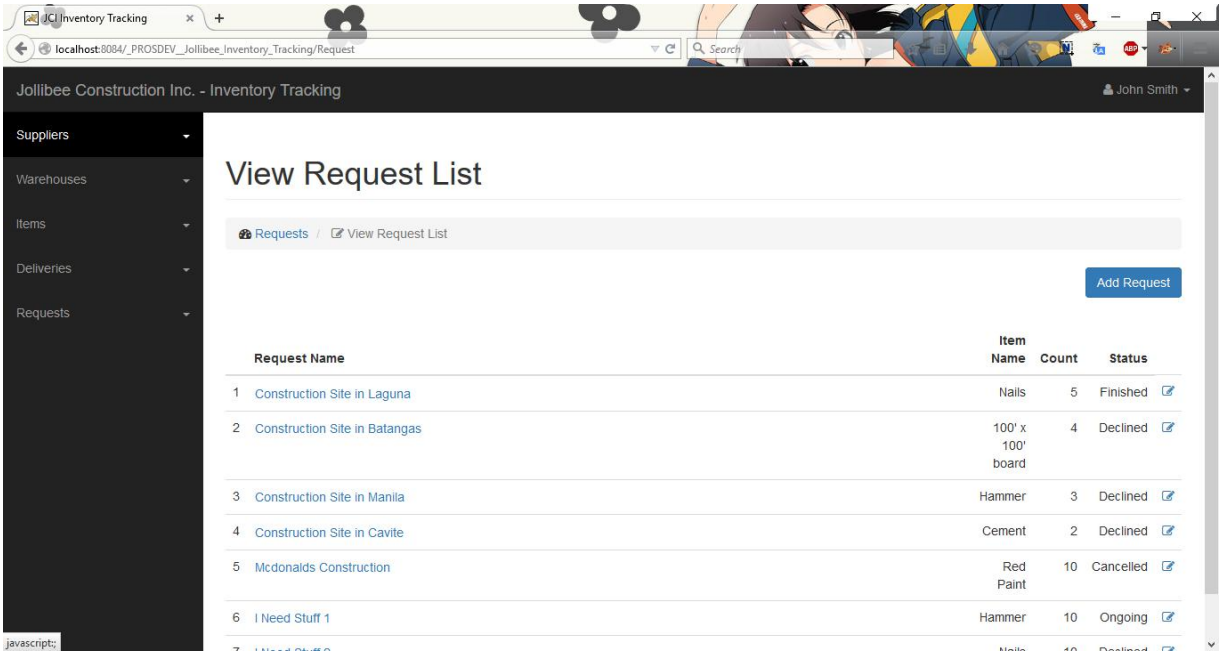
The second major part of a Bootstrap Admin page is the side bar. This is where you place all links the current user can access based on his/her role. Its content can be found at leftbar.html and its structure looks like the following:

```
<!-- Sidebar Menu Items - These collapse to the responsive navigation menu
on small screens -->

<div class="collapse navbar-collapse navbar-ex1-collapse">

      <ul class="nav navbar-nav side-nav">

            <li>

                  <a data-target="#suppliersNav" data-toggle="collapse"
            href="javascript:;">

                  Suppliers

                  <b class="caret" style="float: right; margin-top:
10px;"></b>

                  </a>

                  <ul id="suppliersNav" class="collapse">

                        <li>

                              <a href="Supplier"><i class="fa fa-fw fa-
file"></i> View Suppliers</a>

                              <a href="addsupplier.jsp"><i class="fa fa-fw
fa-file"></i> Add Supplier</a>

                        </li>

                   </ul>

           </li>

           <li>

                  <a href="bootstrap-grid1.html"><i class="fa fa-fw fa-
wrench"></i> Bootstrap Grid 1</a>

           </li>

     </ul>

</div>
```

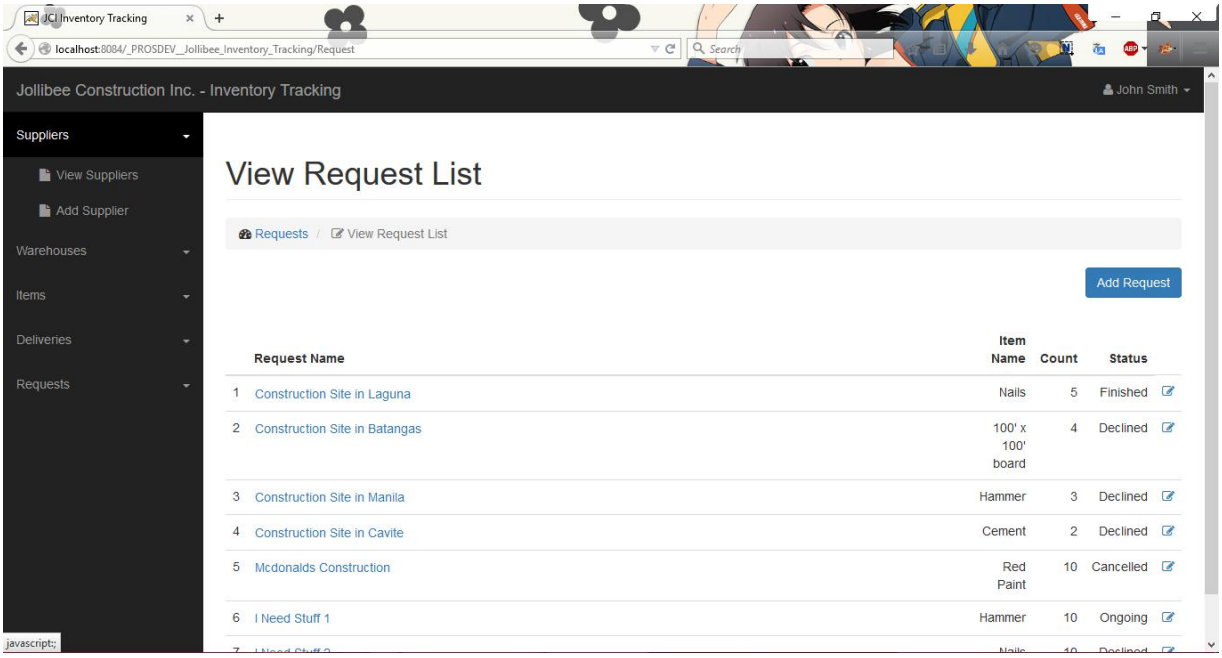
This section starts with the following tags: the `<div class="collapse navbar-collapse navbar-ex1-collapse">` and `</div>` with the `<ul class="nav navbar-nav side-nav">` and `</ul>` tag. Then, inside these tags you can insert either a submenu or a page link. A submenu is a group containing page links which have a similar function. Its markup looks like this:

```
<li>
      <a data-target="#suppliersNav" data-toggle="collapse"
href="javascript:;">
      Suppliers
            <b class="caret" style="float: right; margin-top: 10px;"></b>
      </a>
      <ul id="suppliersNav" class="collapse">
            <li>
                  <a href="Supplier"><i class="fa fa-fw fa-file"></i> View
Suppliers</a>
                  <a href="addsupplier.jsp"><i class="fa fa-fw fa-
file"></i> Add Supplier</a>
            </li>
      </ul>
</li>
```

The submenu contains two parts: the header and the body. The header is the submenu's label. The example's label is the one highlighted in the box below:



And the <ul> tag found after the header is the submenu's body. In the given example, when you click on the header, this happens:



Which means that it shows the contents of the <ul> tag. Remember: this will only work if you set your header's 'data-target' tag value as the class or id of the submenu's body. In this case, the 'data-target' tag's value is '#suppliersNav', which points to the <ul> id value of 'suppliersNav'. If you don't set the right tag values, the submenu will not work through its contents not displayed properly.

Finally, after the navigation part of the page, it's the actual body of the page. This is where you place all the page's contents. Before putting all pages here, make sure that everything you place as the page's actual content is enclosed in a `<div id="page-wrapper">` followed by a `<div class="container-fluid">`. Then, for each subsection of the body, wrap it around a `<div class="row">`.

Hmm…doesn't the class 'row' remind you of tables? Well, let me introduce to you Bootstrap's grid system! Said grid system provides an easy way for elements in a page to appear as they are without going through too much trial-and-error to position them right. For the grid system, make sure that the div is inside a . Then, that div must be inside a div with a '.container' or '.container-fluid' class. Then, for any element inside said div, use either a '.col-xs', '.col-sm', '.col-md' or '.col-lg' tag followed by a hyphen and a number between 1 to 12. Bootstrap divides the page into 12 equal parts under the '.container' and '.container-fluid' tags. For our project, use '.col-md' since it's the average size of screens.

For all pages in the project, it is composed of two important elements: the header and the breadcrumbs. Combined, they both look like this:

```
<!-- Page Heading -->
        <div class="row">
            <div class="col-lg-12">
                <h1 class="page-header">
                    View Request List
                </h1>
                <ol class="breadcrumb">
                    <li>
                        <i class="fa fa-dashboard"></i>  <a href="blank-
page.html">Requests</a>
                    </li>
                    <li class="active">
                        <i class="fa fa-edit"></i> View Request List
                    </li>
                </ol>
            </div>
        </div>
        <!-- /.row -->
```

The header alone looks like this:

```
<h1 class="page-header">
      View Request List
</h1>
```

While the breadcrumb looks like this:

```
<ol class=”breadcrumb”>
      <li>
            <i class=”fa fa-dashboard”></i> <a href=”blank-
page.html”>Requests</a>
      </li>
      <li class=”active”>
            <i class=”fa fa-dashboard”></i> View Requests
      </li>
</ol>
```

A breadcrumb is designed for a user to identify where in the system he/she is. The entries before the <li class=”active”> tag will show where the user has been within a subgroup. In the example above, the dashboard will then appear as Requests > **View Requests** where ‘Requests’ will be linked to the ‘Requests’ homepage while ‘View Requests’ is the current page the user is on.

After creating the heading and the breadcrumbs, you may now proceed to create the actual page content. In this project, forms and tables are important.

For forms, each element can have a label to go with it. For example:

```
<form class="add-supplier-form" id="add-item-form”>
      <div class="form-group">
            <label>Name</label>
            <input id="name" class="form-control" name="name" required>
      </div><!-- end of .form-group -->
      <input type=”submit” class=”btn btn-primary” value=”Submit”>
</form>
```

You enclose them within an outer div with the class “form-group” to align them accordingly. Then, for the input element, always include the “form-control” to make sure that the element will appear nicely along with the page. To add a label in the input element, add ‘ placeholder=”placeholder” ’ within its tag.

An example for tables:

```
<div class=”table-responsive”>
      <table class=”table table-hover”>
            <thead>
                  <tr>
                        <th>#</th>
                        <th>Name</th>
                  </tr>
            </thead>
            <tbody>
                  <tr>
                        <th>1</th>
                        <th>makoto37</th>
                  </tr>
                  <tr>
                        <th>1</th>
                        <th>makoto38</th>
                  </tr>
                  <tr>
                        <th>1</th>
                        <th>makoto39</th>
                  </tr>
            </tbody>
      </table>
</div>
```

When declaring a table, start with declaring then declare the table with the class ‘table’. The ‘table-hover’ tag lets you highlight the current row under the mouse pointer.

Notice the <i> tags like this:

```
<i class=”fa fa-dashboard”></i>
```

They're actually little icons. How come they have pictures? Bootstrap combined with font-awesome icons are the culprit.

To insert an icon, you have to find the right one in the pdf named 'bootstrap-icons.pdf. Then, add the code inside the tag of <i code="fa <icon code>"></i>. Note: insert the code with the one starting with 'fa-'.

To insert an icon, you have to find the right one at fontawesome.bootstrapcheatsheets.com then add it to <i class="fa <icon code>"></i>.

Finally…modals!

Modals are what we call pop-ups. In any HTML page, add the code if you have to use the modal:

```
<div class="modal fade" id="modal" tabindex="-1" role="dialog" aria-
labelledby="messageModal">
      <div class="modal-dialog" role="document">
            <div class="modal-content">
                  <div class="modal-body">
                        <h4 class="modal-title" id="modal-message"></h4>
                  </div>
                  <div class="modal-footer">
                  </div>
            </div>
      </div>
</div>
```

Then, for example, you want the modal to appear when a certain element in the page is clicked, say in a icon, add this:

```
<a id="activate-modal" class="activate-modal" data-toggle="modal"
      data-target="#modal" data-verdict="question">
            <i class="fa fa-question"></i>
</a>
```

So that when you click that element, the modal will appear. But then, the modal is empty. For this project, I filled the modal using javascript.

Add the following code to js to let the modal fill and empty itself:

```
$('#modal').on('show.bs.modal', function(event){
      var trigger = $(event.relatedTarget);
      var verdict = trigger.data('verdict');
      var modal = $(this);
      if(verdict==='question'){
            modal.find('.modal-title').text("Hello world!");
            modal.find('.modal-footer').append('<button type="button"
class="btn btn-primary" data-dismiss="modal">Hi to you too!</button>');
      }
});
$('#modal').on('hide.bs.modal', function(event){
      var modal = $(this);
      modal.find('.modal-footer').empty();
});
```

The first set of js code with 'show.bs.modal' triggers in between the event of triggering the modal to show and the event of the modal appearing. 'hide.bs.modal' triggers in between the event of triggering the modal to hide and the event of the modal hiding. Inside the event of 'show.bs.modal' is where you place the code to fill the modal while you place the code to empty the modal during 'hide.bs.modal'. The checking in 'show.bs.modal' is done through the value of the 'data-verdict' tag in the element where the modal can be triggered. Its value is then retrieved using data('verdict'), then is used in the if compare to determine how to fill the modal. In 'hide.bs.modal', the modal.find('.modal-footer').empty() is used to empty the modal's footer.

But after everything, there's a final imported page called 'footer.html'. It contains the following:

```
                              </div>
                              <!--/.container-fluid-->
                        </div>
                        <!--/#page-wrapper-->
                  </div>
                  <!--/#wrapper-->
            <!--jQuery-->
            <script src="js/jquery.js"></script>
            <!-- Bootstrap Core Javascript -->
            <script src="js/bootstrap.min.js"></script>
            <!-- Custom Javascript-->
            <script src="js/customjs.js"></script>
      </body>
```

Basically, footer.html just contains the closing tags for all the main pages. It also includes all the important javascript pages.

Yay! Bootstrap admin front-end tutorial done! For more information, see getbootstrap.com for more information regarding all things bootstrap.