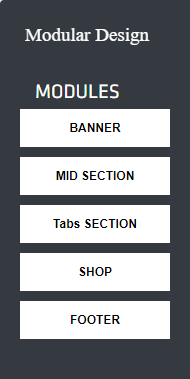
AMD Modular Documentation

Introduction

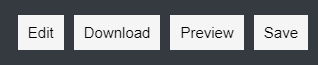
This project is about make the webpage in Modular Design concept. In this Project we have made different modules of websites. We had divided the content into different sections like the Banner, Mid-section, Tabs-section, Shop and Footer.



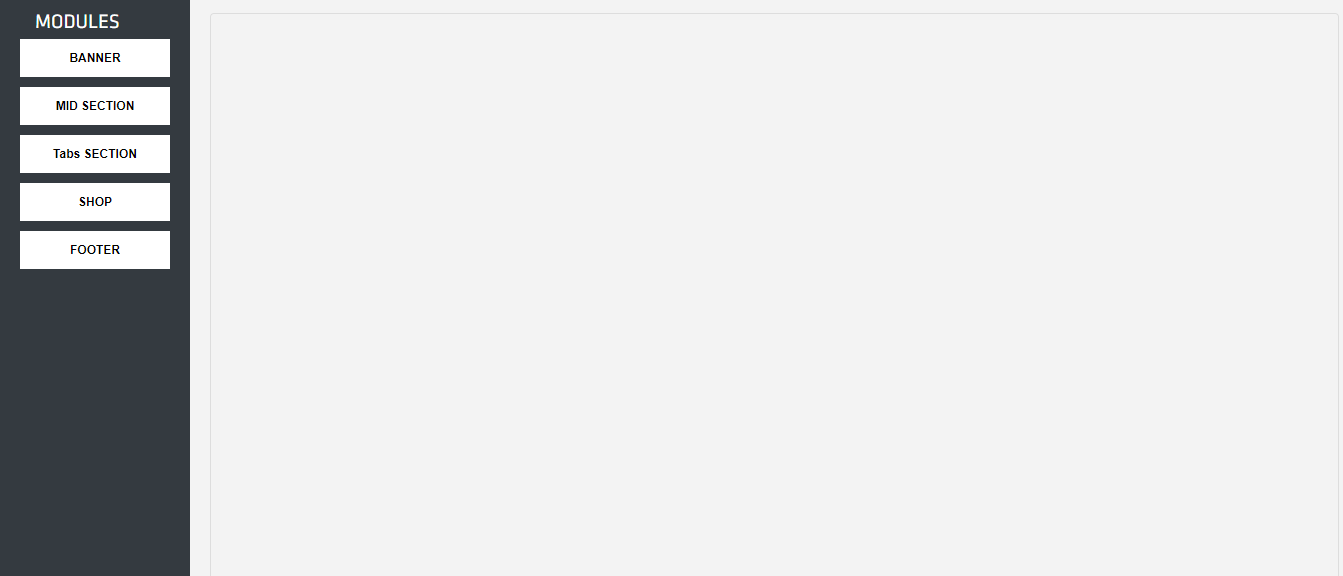
display: none

In the above figure we make a div and under that div we take two div one for the sidebar and another for the container where the element is going to be dropped. In the sidebar we make a div named “BANNER” in which we take another div for the content of the banner section. Since it is not displayed in the sidebar div because we hide the content of the banner section with **“display” “none”** property. It will displayed when the element is dropped into the droppable container. Similarly, we do with all different modules of the webpage.

As this is a modular website so every part of the webpage is in made with modular concept. In this we also added the Edit, Download, Preview and Save buttons.

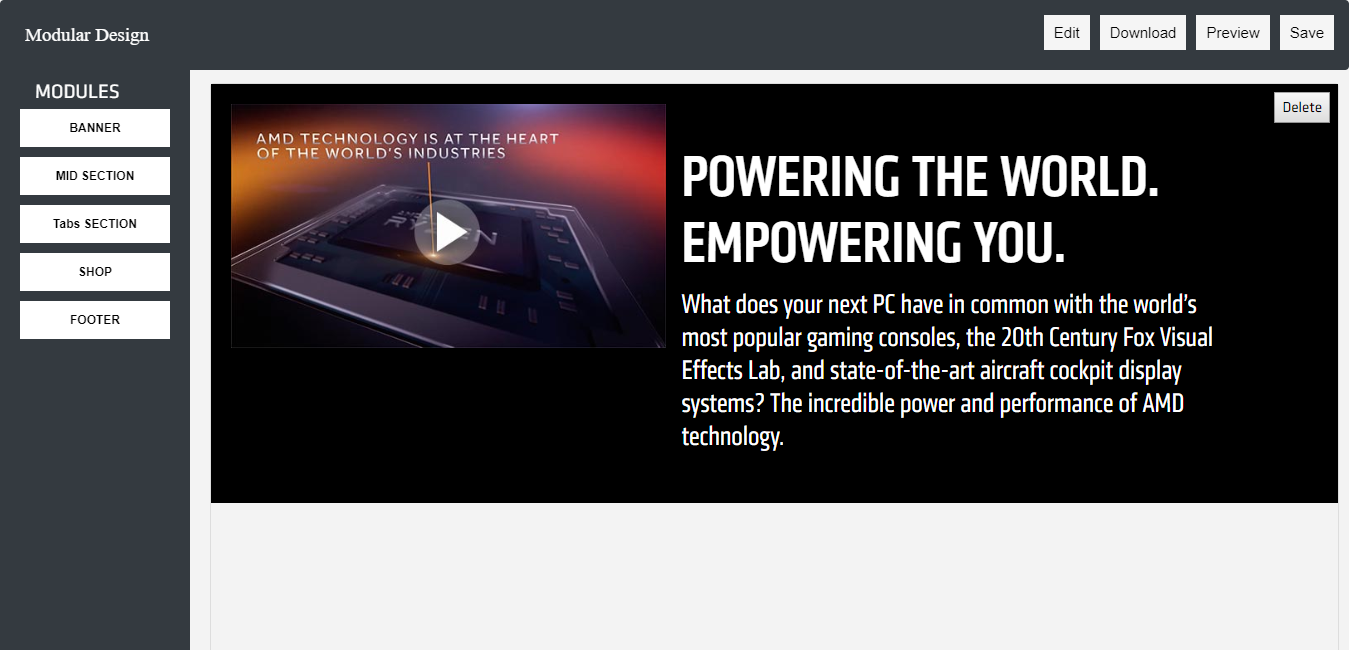


There is also a container where we drag the modules into it.



Working

In this project we have created different modules. User can take any module of his/her choice and drag from the sidebar containing different modules and drop in the container.



**Drag and Drop function**

In the above figure we have drag the midsection module and drag into the container. For making the element Draggable we used the JQuery UI function **“.draggable ()”** to make the element draggable.

$("#neww4,#neww8,#neww14,#neww13,#neww6,#neww7,#neww144,#neww22").draggable({

    helper:"clone",cursor: "pointer",});

In the code we just provide the id’s of the element which we want to draggable.

After dragging the element we drop into the container with the help of again JQuery UI function **“.droppable ()”** to make the element droppable into it.

$("#new1").droppable({

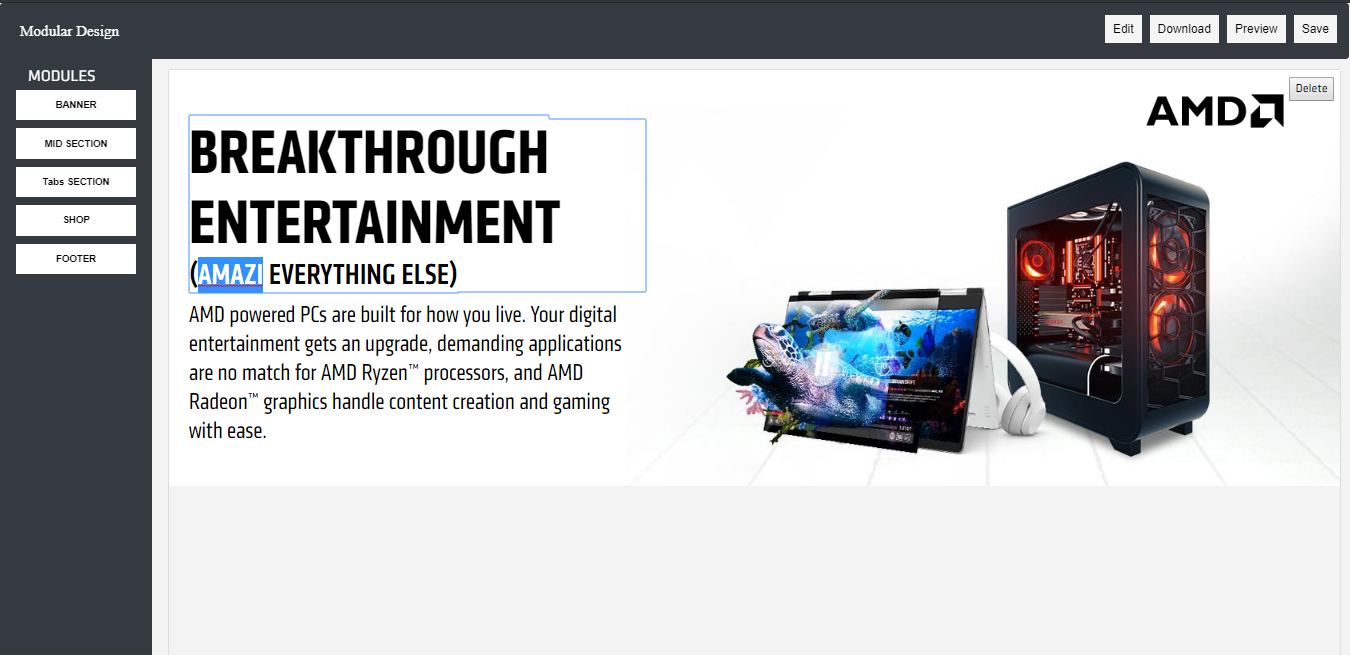
    drop: function( *event*, *ui*) {

      ui.draggable.clone().appendTo("#new1");

In the above code we make the container with id “new1” droppable with the function **“.droppable ()”**. In that we give the id of the container and also make the event “drop” to make the droppable function to be occurred. In the drop event we used the function “appendTo” to whatever element we want to drag is appending to the container.

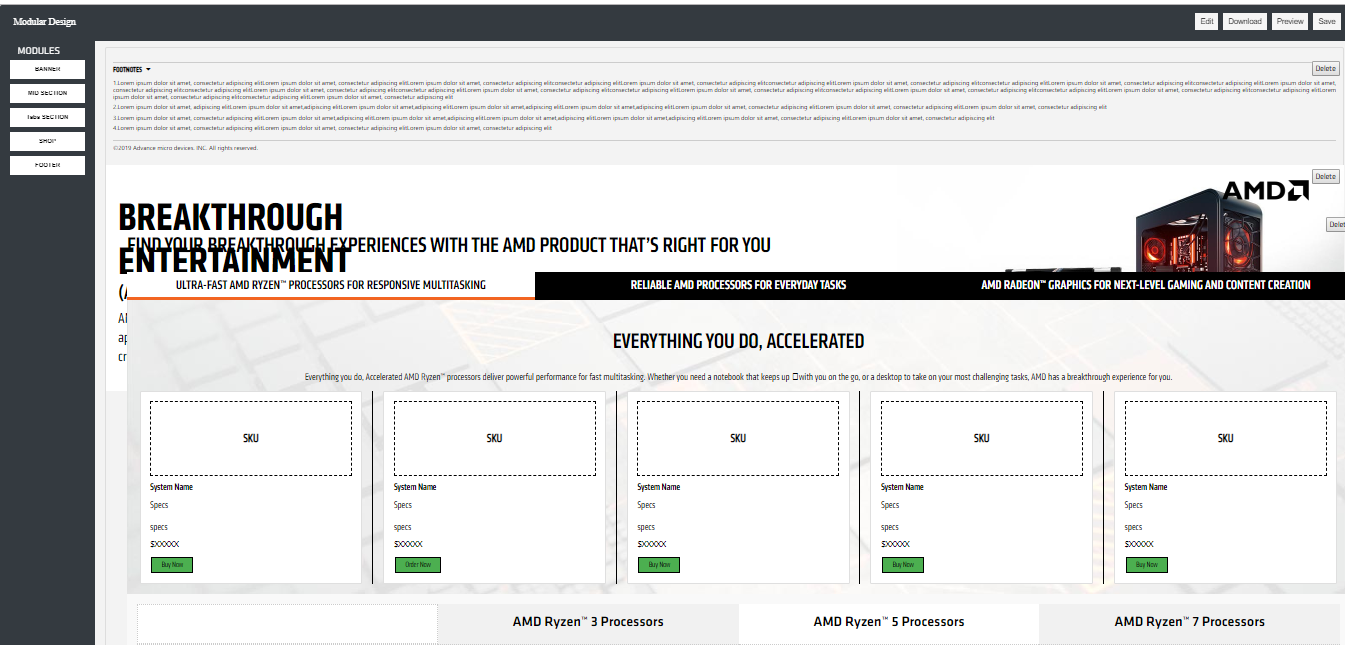
**Make the content to be editable**

Similarly, we make the every module draggable and it will be droppable into the container. After the dropping the element dropped into the container. Now, if user want to change something like the content then they can change the content like to change the heading, paragraph etc. For make the element editable we used the method **“contenteditable=true”** for each part of webpage which is supposed to be editable. Just like in the below figure we can change the module heading.



Content of the module is editable

**Make the modules sortable**



In the above figure we make the modules to be sortable which means that we can able to change the position such that if we want the banner section after mid-section then we have to just grab the module and drop it after the mid-section. Then the banner section will be set after the mid-section.

For that we just use the JQuery function **“.sortable ()”** to make the modules sortable.

 $("#new1").sortable({

        helper: 'clone',

        receive: function (*event*, *ui*) {

        ui.item.remove();

        }

});

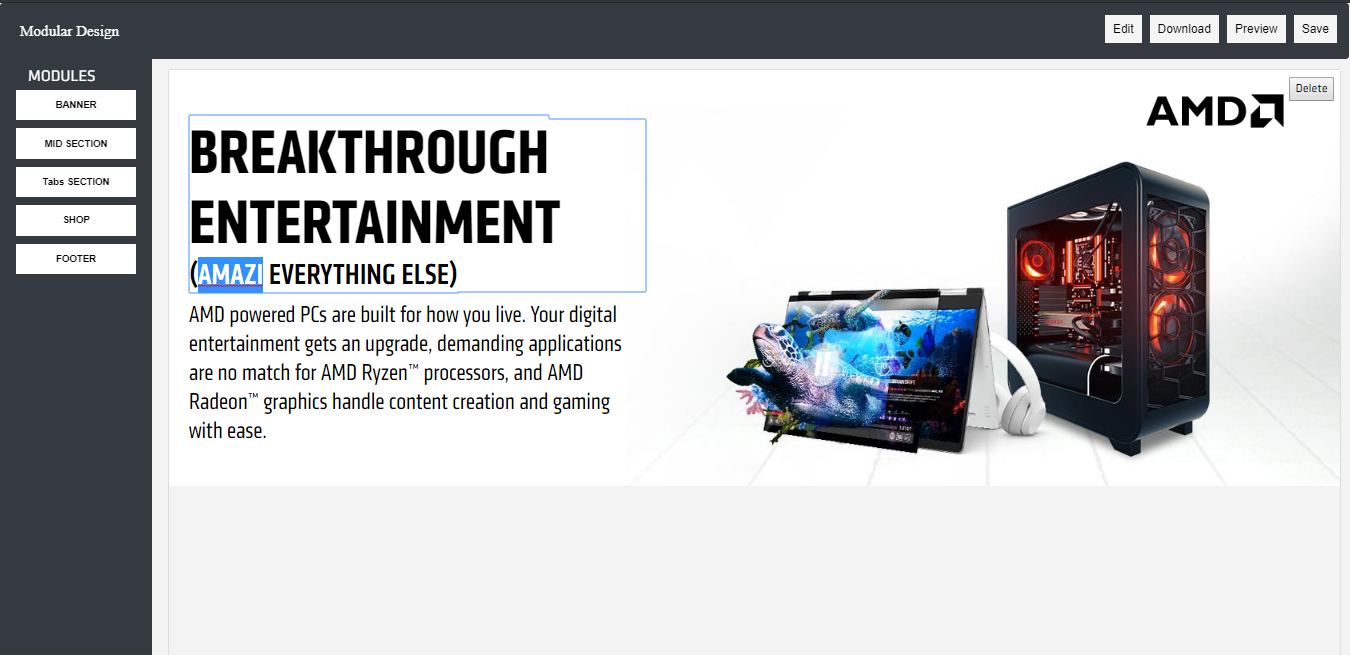
In the above code we have taken a **.sortable ()** function in which we provide the id **“new1”** of the container such that every element of the container sortable. After that we make an event such that whenever we grab the container and drop it, then the event will remove the duplicate element created when we drop it into the container.

**Save Button**

After when the user will satisfying with the content which he/she had made then the user will save the content with the **Save** button.

**SAVE BUTTON**

If user want to do changes something they can change the paragraph, heading etc. of that module and save the changes using the **Save** button.



function saveEdits() {

var editElem = document.getElementById("new1");

var userVersion = editElem.innerHTML;

localStorage.userEdits = userVersion;

}

function checkEdits() {

if(localStorage.userEdits!=null)

document.getElementById("new1").innerHTML = localStorage.userEdits;

}

In this code we have taken a function named **saveEdits ().** In which we have taken a variable **editElem,** in this variable we get the element which is going to be editable by **document.getlementById ()**. After that we have taken another variable **userVersion** where all the editable content where going to be stored locally into a variable.

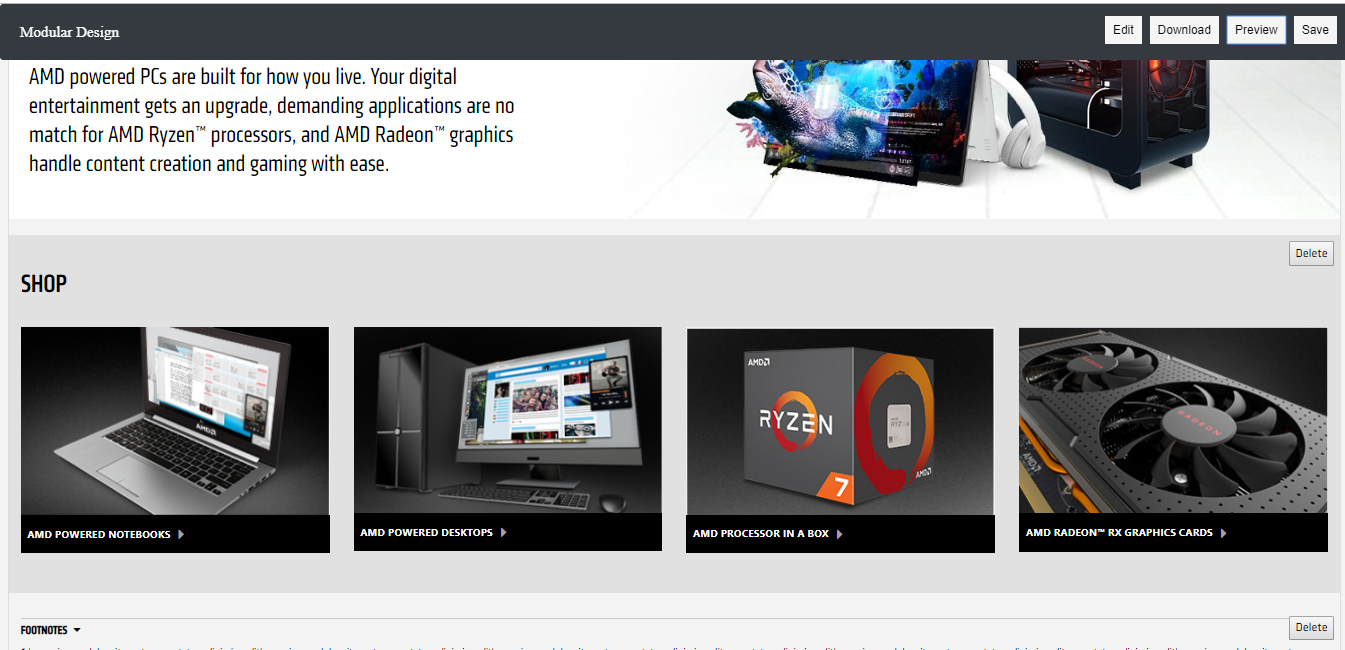
Another function **checkEdits ()** such that when the editable element is none then all the content will be same as the previous. And when the editable element is there then it will save the change which we had made.

<body class="edit builder" onload="checkEdits()" >

We have to attached **onload=”chechEdits ()”** to the body tag. Then the function of the save button will work.

**Preview Button**

Similarly they can do with all other modules. They just have to drag the modules and drop in the container and do changes if they want and save the content. After drag the container into its respective positions and did changes in the content. Now they can preview their webpage using the **Preview** button. As with the help of preview buttons they can see what the website look like in actual after make the changes.



Body will be shift towards left.

$("#pre").click(function (){

    $(".sidebar-nav").css("display","none");

    $("body").css("margin-left","0px");

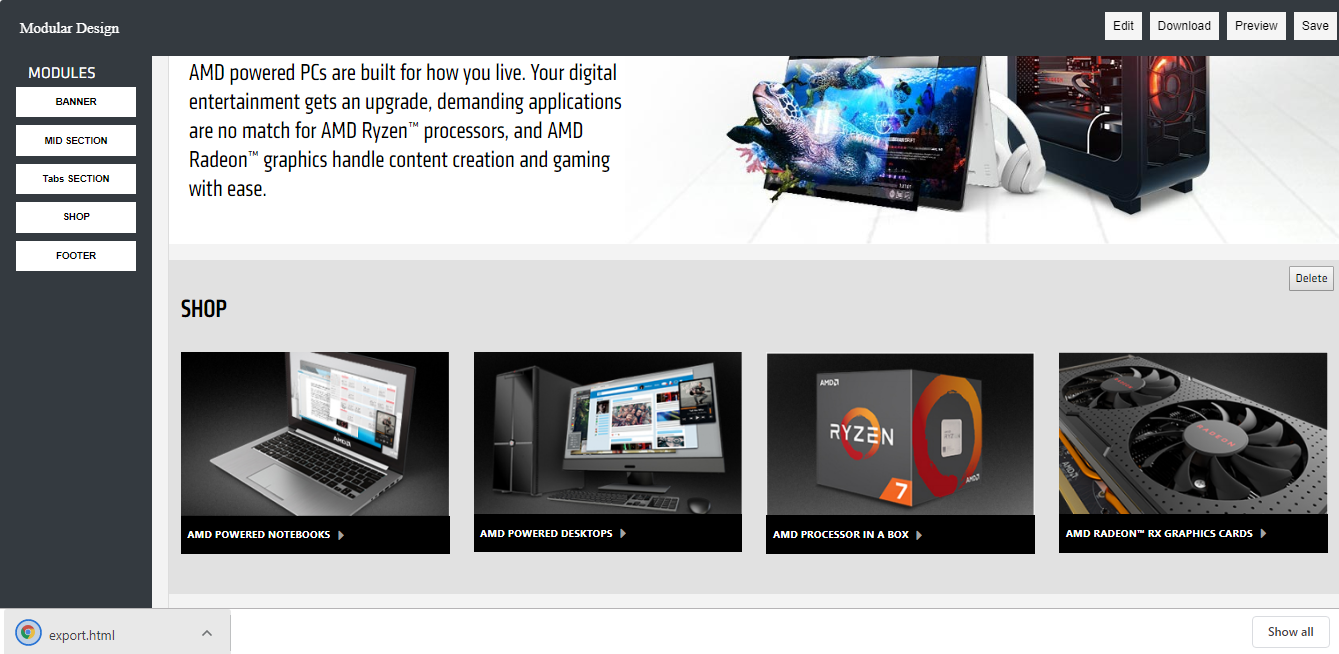
    $(".navbar").css("margin-left","0px");

  });

In the above figure when user click on the preview button then the side bar will disappear as we provided the “display” “none” property to the side bar and at the same time body of the webpage will margin-left to “0 px” and navigation bar will also margin-left to “0 px”.

**Download Button**

If they want to add something more in the website they can use the **Edit** button. After satisfying with the changes done in the website. Now user can download the website code with the help of **Download** button.



**Downloaded Material**

We have taken different modules of each section with its own HTML and CSS code such that no conflicts of code between different modules. Like the Banner section has its own code and Mid-Section has its own code and respectively.

function download(*filename*, *exp\_content*) {

 var element = document.createElement('a');

      element.setAttribute('href', 'data:text/html,' + encodeURIComponent(exp\_content));

      element.setAttribute('download', filename);

      element.style.display = 'none';

      document.body.appendChild(element);

      element.click();

      document.body.removeChild(element);

                }

*// Start file download.*

document.getElementById("download").addEventListener("click", function(){

   var editableElements = document.querySelectorAll("[contenteditable=true]");

            for (var i = 0; i < editableElements.length; i++) {

              editableElements[i].setAttribute("contentEditable", false);

            }

     $(".header, .topnav, #about, #services").css("display","block");

        var csslinks = "<link rel='stylesheet' type='text/css' href='https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css'>            <link rel='stylesheet' href='https://cdnjs.cloudflare.com/ajax/libs/bootswatch/3.3.7/darkly/bootstrap.min.css'>";

var exp\_content = csslinks + style + content + script\_start +script ;

      var filename = "export.html";

      download(filename, exp\_content);

                }, false);

In this code we had taken a function named download and passing two parameters namely filename which provide the name of the file when you click the **Download** button to the function and a content which provide the content which going to be downloaded. After that we had created a variable name element in which we have pass the anchor tag into it. Then we will use the **set Attribute () function** in which we push the “href”, which provide the link and “data: text/html” which provide the content and lastly it will encoded and compressed into a downloaded file.

Again, we have taken a variable **csslinks** in which we put all the css links into it. Similarly, we have taken another variable **style** where we put all the CSS styles into it. Another variable exp\_content which we had pass in our function. In which we concatenate all the variables which we are using in our download function.