**ANGULAR JS DOCUMENT**

**Technologies Used:**

FrontEnd: Angular.Js

MiddleWare:Node.js+Express

Backend: Mongodb Database,Mongoose

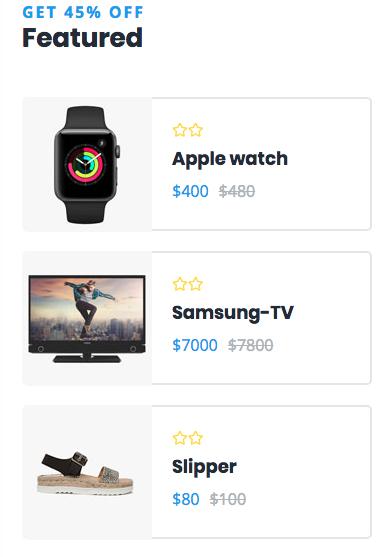
**Functional Description**

**1)Landing Page Component**

Created three components for displaying featured product, bestseller product and offer ad.

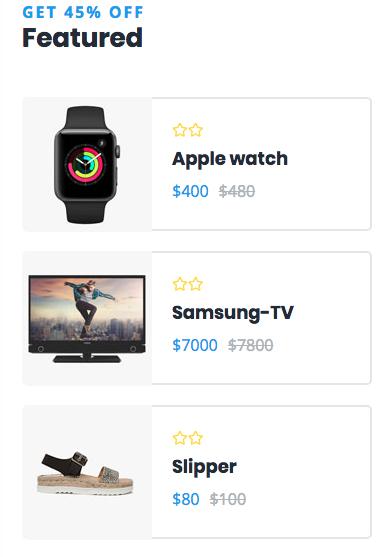
a) Featured Product Component:

Featured product lists are coming from mongodb database. The table consists of image, rating, title, selling price and discount price.



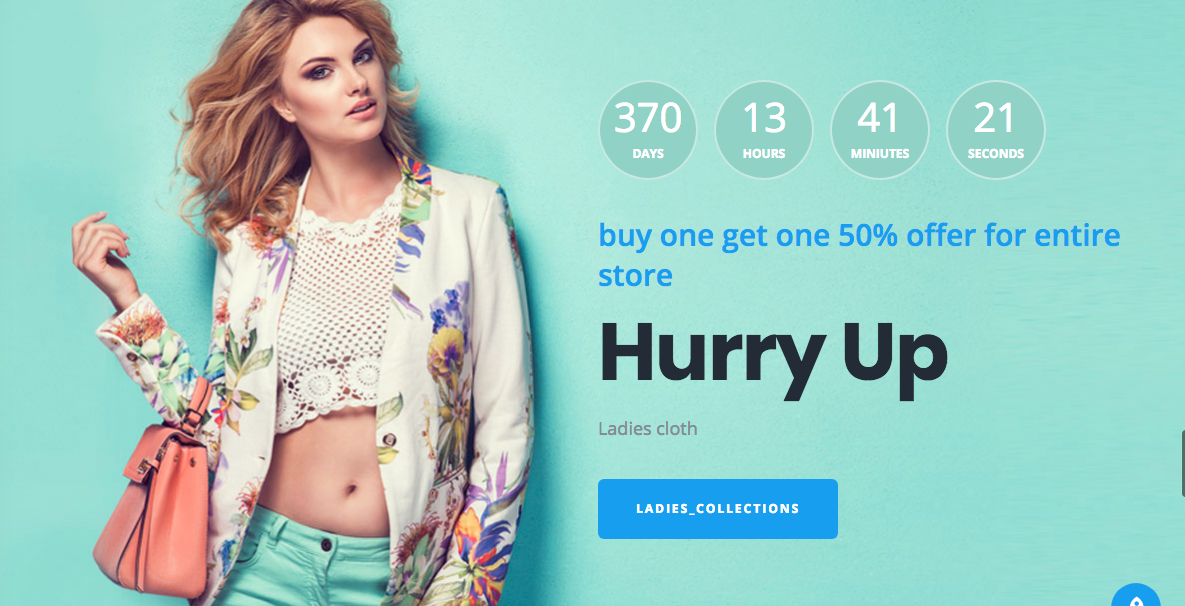
b) **Best Seller Product:**

Data are coming from database consists of image, rating, name of product using the rest call this products are displaying in GUI.



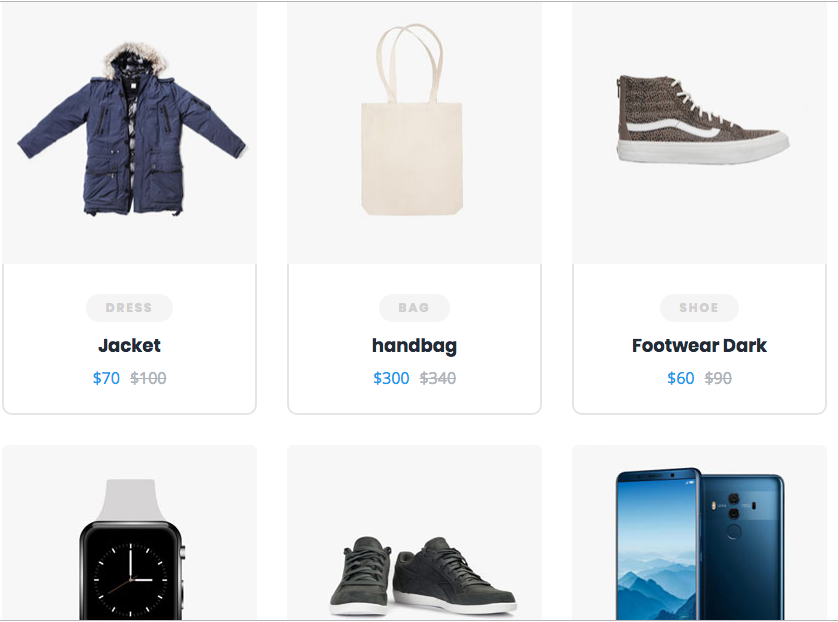
c) **Ladies Cloth Advertisement**

This advertisement layer consists of countdown, title, description, image and header. All these are coming from database.



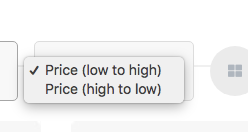
**2) Search Component**

1. **Display the list of products in search**- these data are coming from database.

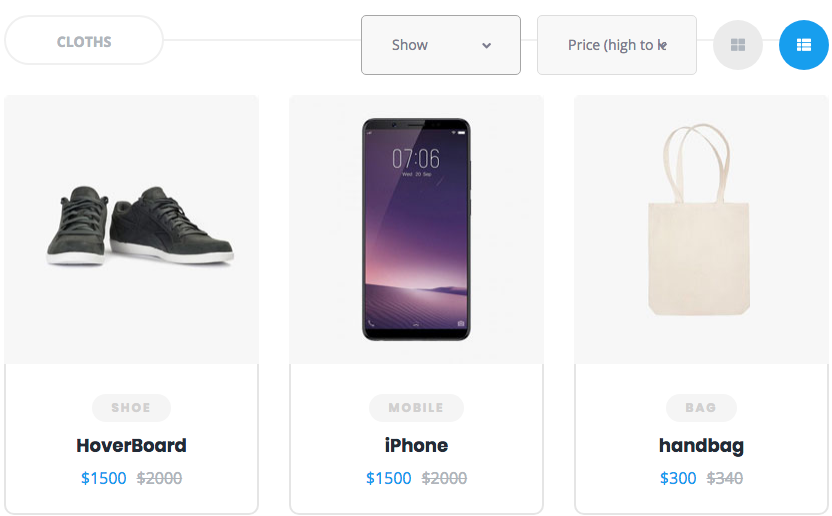


Search table includes name, category, discount price, selling price, image and id.

1. **Sorting the product based on Price**

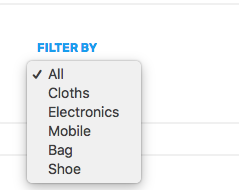


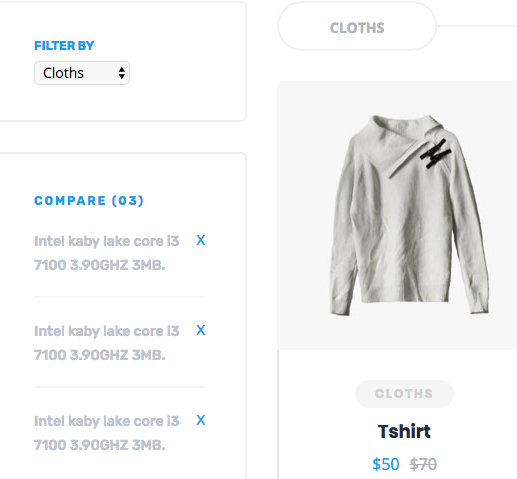
Here the product displayed based on price range. I created the click event on select option. I did the sorting in client side code with comparator.



1. **Filter the product based on the category**

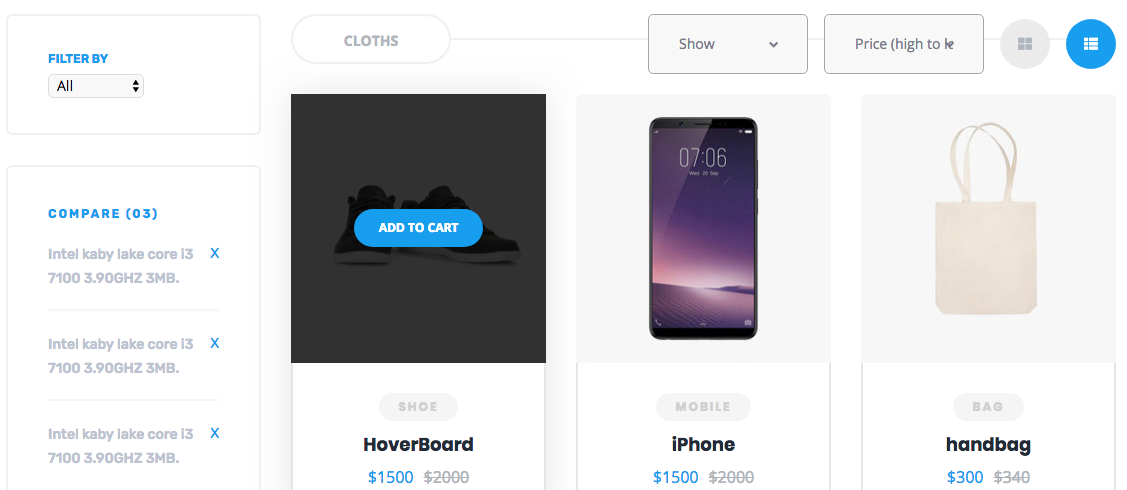
Applying the filter on select option to display the list with category.





1. **Removing filter from select option**

When click all option it should display the all products from the list.

****

**Technical Document:**

**FRONT END CODE**  
CREATE COMPONENTS FOR FEATURED,BEST SELLER AND OFFERAD

**FEATURED SERVICE.TS FILE**

public getfeaturedProduct():Observable<any>{

//console.log("service layer is called");

const endpoint=AppConfig.FEATURED\_ENDPOINT;

console.log("service layer is called");

console.log(endpoint);

return this.http.get(endpoint);

}

**FEATURED COMPONENTS.TS FILE**

import { Component, OnInit } from '@angular/core';

import { AppConfig } from 'src/app/config/app.config';

import {FeaturedService} from 'src/app/services/featured.service';

import {ActivatedRoute} from '@angular/router';

@Component({

selector: 'app-featured',

templateUrl: './featured.component.html',

styleUrls: ['./featured.component.css']

})

export class FeaturedComponent implements OnInit {

public featuredProduct:[]=[];

public baseURI:String="";

constructor(private route:ActivatedRoute,private featuredService:FeaturedService) { }

ngOnInit() {

this.featuredService.getfeaturedProduct().subscribe((data)=>{

for (var key in data) {

this.featuredProduct[key]=data[key];

}

console.log(this.featuredProduct);

this.baseURI=AppConfig.BASE\_ENDPOINT;

});

}

APPCONFIG

public static FEATURED\_ENDPOINT = AppConfig.BASE\_ENDPOINT + '/admin/featured';

NODE JS CODE

CREATE CONNECTION WITH MONGODB DATABASE

Mongo-utils.js

var mongoose = require('mongoose');

/\*\*

\* When you call this method then it would

\* connect you to the database

\*/

module.exports=function() {

    var conn = mongoose.connection;

    conn.on('error', console.error);

    conn.once('open', function() {

        //  Create your schemas and models here.

        console.log("Opening connection to database!");

    });

    mongoose.connect('mongodb://localhost:27017/shopping\_cart\_db');

};

CREATE SCHEMA – (ENTITY.js)

CREATE SCHEMA UNDER MODEL FOLDER

Featured-entity.js

var mongoose=require('mongoose');

var FeaturedSchema=new mongoose.Schema({

pid:{type:String,required:true,unique:true},

imageUrl:{type:String},

title:{type:String},

sprice:{type:String},

dprice:{type:String},

rating:{type:String},

doe:{type:Date},

dom:{type:Date},

userid: { type: String}

},{collection:'featured\_collections'});

//save the data with date

FeaturedSchema.pre('save',function(next){

var currentDate=new Date();

this.dom=currentDate;

//if not exist

if(!this.doe){

this.doe=currentDate;

}

next();

});

var FeaturedEntity=mongoose.model('Featured',FeaturedSchema);

module.exports=FeaturedEntity;

DATAPUSHER TO PUSH DATA INTO DATABASE

var FeaturedEntity=require('../model/featured-entity')

module.exports=function(){

var featured1 =new FeaturedEntity();

featured1.pid="100";

featured1.imageUrl="assets/img/product-small/18.jpg ";

featured1.rating="2";

featured1.title="Apple watch";

featured1.sprice="$400";

featured1.dprice="$480";

featured1.save(function(err){

if(!err)

console.log("save in database");

else

console.log("");

});

var featured2 =new FeaturedEntity();

featured2.pid="500";

featured2.imageUrl="assets/img/product-small/17.jpg";

featured2.rating="7";

featured2.title="Samsung-TV";

featured2.sprice="$7000";

featured2.dprice="$7800";

featured2.save(function(err){

if(!err)

console.log("pushed");

CREATE CONTROLLER FOR EACH COMPONETS NAME ENDS WITH .js files

fs=require('fs');

var FeaturedEntity=require('../../model/featured-entity')

var ImageFolderPath=require('../../config/image-folder-path')

module.exports.getFeaturedProduct=(req,res)=>{

//var ppid=req.query.pid;

FeaturedEntity.find({ },(err,data)=>{

if(!err){

console.log(data);

return res.status(200).json(data);

}else{

return res.status(400).send({

status:"fail",

message:"not found " +ppid

});

}

});

}

module.exports.save =(req,res)=>{

console.log("save the feature details in db");

var featured = req.body;

console.log(featured);

var featuredEntity=new FeaturedEntity();

let pphoto=req.files.photo;

var imgtxt=pphoto.name.substring(pphoto.name.length-4);

var pphotoName=featured.pid+" "+imgtxt;

featuredEntity.imageUrl=pphotoName;

console.log(appRoot+"/"+featuredEntity.imageUrl);

pphoto.mv(appRoot+"/"+ImageFolderPath.FEATURED\_IMAGES+"/"+featuredEntity.imageUrl,function(err){

if(err){

console.log("err");

return res.status(500).json({status:"fail",message:"images are not saved"})

}

else{

featuredEntity.userid=featured.userid;

featuredEntity.save(err=>{

if(err){

console.log(err);

res.status(200).json({status:"fail",message:"Sorry!! not able to save in database"});

}else{

res.status(200).json({status:"success",message:"saved in database"})

var FeaturedController=require('../controller/admin/featured-controller')

module.exports=function(endPoint){

//endPoint.get("/admin/featured",FeaturedController.save);

endPoint.get("/admin/featured",FeaturedController.getFeaturedProduct);

}

SPECIFY ENDPOINT IN ROUTER ANS ALSO CONFIGURE APPCONFIG.js file

APPCONFIG.js folder under src->app->config->AppConfig

WRITE BELOW CODE IN SERVER.JS FILE TO DATABASE

require('./rest-api/utils/featured-data-pusher')();

require('./rest-api/utils/bestseller-data-pusher')();

var datapusher=require('./rest-api/utils/ladiescloths-data-pusher');

datapusher();

WRITE BELOW CODE IN SERVER.JS FILE -CONNECT TO ROUTER

require('./rest-api/router/featured-router')(endPoint);

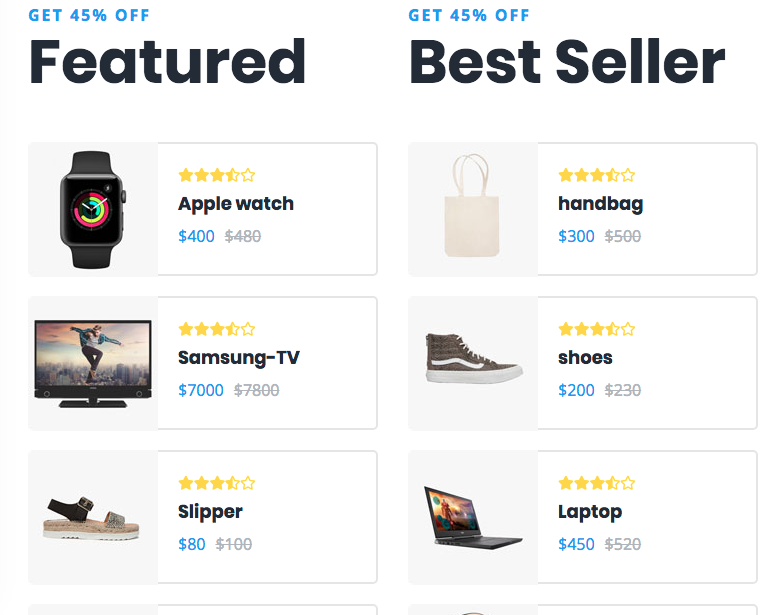
require('./rest-api/router/bestseller-router')(endPoint);

require('./rest-api/router/ladiescloths-router')(endPoint);

HOW TO BRING DATABASE DATA TO HTML PAGE?

<a href=""><h4 class="title">{{featuredProduct[0].title}}</h4></a>

<div class="price"><span class="sprice">{{featuredProduct[0].sprice}}</span> <del class="dprice">{{featuredProduct[0].dprice}}</del>



LADIES CLOTH SECTION

DATABASE CODE

SCHEMA

var mongoose=require('mongoose');

var LadiesclothsSchema=new mongoose.Schema({

pid:{type:String,required:true,unique:true},

subtitle:{type:String},

title:{type:String},

description:{type:String},

userid:{type:String},

doe:{type:Date},

dom:{type:Date}

},{collection:'ladiescloths\_collections'});

LadiesclothsSchema.pre('save',function(next){

console.log("in the entity")

var currentDate=new Date();

this.dom=currentDate;

if(!this.doe){

this.doe=currentDate;

}

next();

});

var LadiesclothsEntity=mongoose.model('Ladiescloths',LadiesclothsSchema);

module.exports=LadiesclothsEntity;

DATAPUSHER

var LadiesclothsEntity=require('../model/ladiescloths-entity')

module.exports=function(){

var ladiescloths =new LadiesclothsEntity();

ladiescloths.pid="100";

ladiescloths.subtitle="buy one get one 50% offer for entire store";

ladiescloths.description="Ladies cloth";

ladiescloths.title="Hurry Up";

ladiescloths.save(function(err){

if(!err) {

console.log("ladies clothes saved in database");

console.log(err);

}

else{

console.log("ladies clothes not save in db");

}

});

}

BRING DATABASE

<span class="subtitle">{{ladiesclothsProduct[0].subtitle}}</span>

<h3 class="title">{{ladiesclothsProduct[0].title}}</h3>

<p>{{ladiesclothsProduct[0].description}}</p>

<a href=" " class="boxed-btn"> ladies\_collections </a>

MIDDLEWARE CODE

module.exports.getLadiesclothsProduct=(req,res)=>{

//var ppid=req.query.pid;

LadiesclothsEntity.find({},(err,data)=>{

if(!err){

console.log(data);

return res.status(200).json(data);

}else{

return res.status(400).send({

status:"fail",

message:"not found "

});

}

});

}

module.exports.save =(req,res)=>{

console.log("save the feature details in db");

var ladiescloths = req.body;

console.log(ladiescloths);

var ladiesclothsEntity=new LadiesclothsEntity();

let pphoto=req.files.photo;

var imgtxt=pphoto.name.substring(pphoto.name.length-4);

var pphotoName=ladiescloths.pid+" "+imgtxt;

ladiesclothsEntity.imageUrl=pphotoName;

console.log(appRoot+"/"+ladiesclothsEntity.imageUrl);

pphoto.mv(appRoot+"/"+ImageFolderPath.LADIES\_CLOTHS\_IMAGES+"/"+ladiesclothsEntity.imageUrl,function(err){

if(err){

console.log("err");

return res.status(500).json({status:"fail",message:"images are not saved"})

}

else{

ladiesclothsEntity.userid=ladiescloths.userid;

ladiesclothsEntity.save(err=>{

if(err){

console.log(err);

res.status(200).json({status:"fail",message:"Sorry!! not able to save in database"});

}else{

res.status(200).json({status:"success",message:"saved in database"});

}

});

}

});

}

ROUTER.JS FILE

var LadiesclothsController=require('../controller/admin/ladiescloths-controller')

module.exports=function(endPoint){

endPoint.get("/admin/ladiescloths",LadiesclothsController.getLadiesclothsProduct);

};

FRONT END CODE

SERVICES.ts

@Injectable({

providedIn: 'root'

})

export class LadiesclothsService {

constructor(private http:HttpClient) { }

public getladiesclothsProduct():Observable<any>{

console.log("from service layer");

const endpoint=AppConfig.LADIES\_CLOTH\_ENDPOINT;

console.log(endpoint);

return this.http.get(endpoint);

}

}

COMPONENT.TS FILE

export class LadiesClothsComponent implements OnInit {

public ladiesclothsProduct:[]=[];

public baseURI:String="";

constructor(private ladiesclothsService:LadiesclothsService) { }

ngOnInit() {

this.ladiesclothsService.getladiesclothsProduct().subscribe((data)=>{

for(var key in data){

this.ladiesclothsProduct[key]=data[key];

}

console.log(this.ladiesclothsProduct);

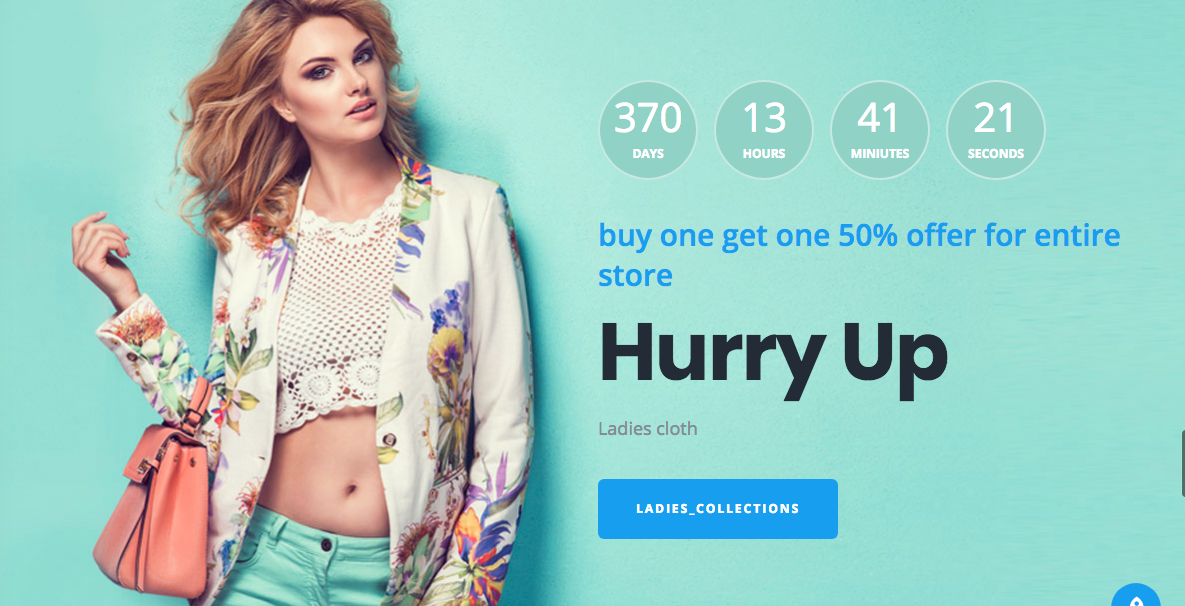
this.baseURI=AppConfig.BASE\_ENDPOINT;

});

}

}

SCREEN SHOTS



**SEARCH COMPONENT:**

**DISPLAY THE LIST OF PRODUCT BASED ON PRICE:**

**Write the below code in component.ts file**

dynamicSort(smethod){

console.log("@@@@@");

console.log(smethod);

if(smethod=="Asc"){

this.searchProduct.sort(function(product1:Search, product2:Search){

let comparison = 0;

if(product1.sprice > product2.sprice){

comparison = -1;

}else if(product1.sprice < product2.sprice){

comparison = 1;

}

return comparison;

}

);

}else{

this.searchProduct.sort(function(product1:Search, product2:Search){

let comparison = 0;

if(product1.sprice < product2.sprice){

comparison = -1;

}else if(product1.sprice > product2.sprice){

comparison = 1;

}

return comparison;

}

);

}

}

getSortOrders(){

console.log(this.sortingMethod);

this.dynamicSort(this.sortingMethod);

}

sorting done with comparator.

Bringing Sorted product to HTML Page :

<select [(ngModel)]="sortingMethod" name="sortingMethod" class="input-field select" (change)="getSortOrders()">

<option value="Asc">Price (low to high)</option>

<option value="Desc">Price (high to low)</option>

</select>

**FILTER THE PRODUCT BASED ON CATEGORY**

**Bringing filter product to HTMl page:**

<select #scategory="ngModel" ngModel name="selectedCategory" (change)="filterProductsByCategory(scategory);">

<option value="">All</option>

<option value="Cloths">Cloths</option>

<option value="Electronics">Electronics</option>

<option value="Mobile">Mobile</option>

<option value="Bag">Bag</option>

<option value="Shoe">Shoe</option>

</select>

TAKING THE BACKUP COPY FROM SEARCH PRODUCT

filterProductsByCategory(filterval:any){

console.log("$$$$$$$$$$$$$$$$ from category");

if(filterval.value==""){

this.searchProduct=this.backupsearchProduct;

}else{

console.log(filterval.value);

console.log(this.searchProduct);

this.searchProduct=this.backupsearchProduct;

this.searchProduct=this.searchProduct.filter((t:Search)=>t.category==filterval.value);

console.log(this.searchProduct);}