ionic start FinalEx ---> Tab

ionic g page home

ionic g page sale

ionic g provider data-service

npm install chart.js –save

IonicPageModule.forChild(MyApp),//lazy looding

App.module.ts

import { NgModule, ErrorHandler } from '@angular/core';

import { BrowserModule } from '@angular/platform-browser';

import { IonicApp, IonicModule, IonicErrorHandler, IonicPageModule } from 'ionic-angular';

import { MyApp } from './app.component';

import { AboutPage } from '../pages/about/about';

import { ContactPage } from '../pages/contact/contact';

import { HomePage } from '../pages/home/home';

import { TabsPage } from '../pages/tabs/tabs';

import { StatusBar } from '@ionic-native/status-bar';

import { SplashScreen } from '@ionic-native/splash-screen';

import { HttpClientModule } from '@angular/common/http';

import { DataServiceProvider } from '../providers/data-service/data-service';

@NgModule({

declarations: [

MyApp,

/\* AboutPage,

ContactPage,

HomePage, \*/

TabsPage

],

imports: [

IonicPageModule.forChild(MyApp),//lazy looding ///// 1

BrowserModule,

IonicModule.forRoot(MyApp),

HttpClientModule

],

bootstrap: [IonicApp],

entryComponents: [

MyApp,

/\* AboutPage,

ContactPage,

HomePage, \*/

TabsPage

],

providers: [

StatusBar,

SplashScreen,

{provide: ErrorHandler, useClass: IonicErrorHandler},

DataServiceProvider

]

})

export class AppModule {}

home.ts

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

import { IonicPage, NavController, NavParams, LoadingController, AlertController } from 'ionic-angular';

import {DataServiceProvider} from '../../providers/data-service/data-service';

/\*\*

\* Generated class for the HomePage page.

\*

\* See https://ionicframework.com/docs/components/#navigation for more info on

\* Ionic pages and navigation.

\*/

@IonicPage()

@Component({

selector: 'page-home',

templateUrl: 'home.html',

})

export class HomePage {

type:any = "email";

Input\_email:any;

Input\_domain:any;

Input\_zip:any;

output\_email:any;

output\_domain:any;

output\_zip:any

constructor(public navCtrl: NavController, public navParams: NavParams,private loadingCtrl: LoadingController,

private DataServiceProvider:DataServiceProvider,private alertCtrl: AlertController) {

}

ionViewDidLoad() {

console.log('ionViewDidLoad HomePage');

}

CheckRex\_email(){

console.log("run === this.DataServiceProvider.PostData({});");

this.presentLoadingCustom();

let run = this.DataServiceProvider.GetData({url:"http://localhost:1880/vaild?input="+this.Input\_email+"&type=email"}).subscribe(value => {

if (value.access) {

console.log("Send value01", value);

//this.listproduct = value;

this.output\_email = value.data.result;

this.loadingView.dismiss();

//return value;

}

else {

console.log("Send value01 else", value);

let errordata = this.DataServiceProvider.HTTP\_errorhandle(value.error);

console.log("errordata",errordata)

this.presentAlert(errordata.title,errordata.message);

this.loadingView.dismiss();

//return value;

}

}, error => { console.error(error.message) },

() => { console.log('Completed subscribe007') }

)

}

CheckRex\_domain(){

console.log("run === this.DataServiceProvider.PostData({});");

this.presentLoadingCustom();

let run = this.DataServiceProvider.GetData({url:"http://localhost:1880/vaild?input="+this.Input\_domain+"&type=domain"}).subscribe(value => {

if (value.access) {

console.log("Send value01", value);

//this.listproduct = value;

this.output\_domain = value.data.result;

this.loadingView.dismiss();

//return value;

}

else {

console.log("Send value01 else", value);

let errordata = this.DataServiceProvider.HTTP\_errorhandle(value.error);

console.log("errordata",errordata)

this.presentAlert(errordata.title,errordata.message);

this.loadingView.dismiss();

//return value;

}

}, error => { console.error(error.message) },

() => { console.log('Completed subscribe007') }

)

}

CheckRex\_zip(){

console.log("run === this.DataServiceProvider.PostData({});");

this.presentLoadingCustom();

let run = this.DataServiceProvider.GetData({url:"http://localhost:1880/vaild?input="+this.Input\_zip+"&type=zip"}).subscribe(value => {

if (value.access) {

console.log("Send value01", value);

//this.listproduct = value;

this.output\_zip = value.data.result;

this.loadingView.dismiss();

//return value;

}

else {

console.log("Send value01 else", value);

let errordata = this.DataServiceProvider.HTTP\_errorhandle(value.error);

console.log("errordata",errordata)

this.presentAlert(errordata.title,errordata.message);

this.loadingView.dismiss();

//return value;

}

}, error => { console.error(error.message) },

() => { console.log('Completed subscribe007') }

)

}

loadingView:any;

presentLoadingCustom() {

this.loadingView = this.loadingCtrl.create({

spinner: 'hide',

content: `

<div class="custom-spinner-container">

<div class="custom-spinner-box"><object data='../../assets/icon/loading.svg' type='image/svg+xml'></object></div>

</div>`,

duration: 15000

});

this.loadingView.onDidDismiss(() => {

console.log('Dismissed loading');

});

this.loadingView.present();

}

presentAlert(title,subTitle) {

let alert = this.alertCtrl.create({

title: title,

subTitle: subTitle,

buttons: ['Dismiss']

});

alert.present();

}

}

Home.html

<!--

Generated template for the HomePage page.

See http://ionicframework.com/docs/components/#navigation for more info on

Ionic pages and navigation.

-->

<ion-header>

<ion-navbar>

<ion-title>home</ion-title>

</ion-navbar>

</ion-header>

<ion-content>

<ion-card>

<ion-item>

<ion-label color="primary" stacked>Email</ion-label>

<ion-input type="email" placeholder="Email Input" [(ngModel)]="Input\_email"></ion-input>

</ion-item>

<button ion-button small (click)="CheckRex\_email()">Check</button>

<h1>{{output\_email}}</h1>

<ion-item>

<ion-label color="primary" stacked>Domain name</ion-label>

<ion-input type="email" placeholder="Domain name" [(ngModel)]="Input\_domain"></ion-input>

</ion-item>

<button ion-button small (click)="CheckRex\_domain()">Check</button>

<h1>{{output\_domain}}</h1>

<ion-item>

<ion-label color="primary" stacked>Zip Code</ion-label>

<ion-input type="email" placeholder="Zip Code" [(ngModel)]="Input\_zip"></ion-input>

</ion-item>

<button ion-button small (click)="CheckRex\_zip()">Check</button>

<h1>{{output\_zip}}</h1>

</ion-card>

</ion-content>

Stock.ts

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

import { IonicPage, NavController, NavParams } from 'ionic-angular';

import {DataServiceProvider} from '../../providers/data-service/data-service'

/\*\*

\* Generated class for the StockPage page.

\*

\* See https://ionicframework.com/docs/components/#navigation for more info on

\* Ionic pages and navigation.

\*/

@IonicPage()

@Component({

selector: 'page-stock',

templateUrl: 'stock.html',

})

export class StockPage {

name:any;

stock\_list = ["2S","A","AAV","ABC","ABICO","ACAP","ACC","ADAM","ADVANC","AEC","AEONTS","AF","AFC","AGE","AH","AHC","AI","AIE","AIRA","AIT","AJ","AJD","AKP","AKR","ALLA","ALT","ALUCON","AMA","AMANAH","AMARIN","AMATA","AMATAV","AMC","ANAN","AOT","AP","APCO","APCS","APURE","AQ","AQUA","ARIP","ARROW","AS","ASEFA","ASIA","ASIAN","ASIMAR","ASK","ASN","ASP","ATP30","AU","AUCT","AYUD","BA","BAFS","BANPU","BAT-3K","BAY","BBL","BCH","BCP","BCPG","BDMS","BEAUTY","BEC","BEM","BFIT","BGT","BH","BIG","BIGC","BIZ","BJC","BJCHI","BKD","BKI","BLA","BLAND","BLISS","BM","BOL","BPP","BR","BROCK","BROOK","BRR","BSBM","BSM","BTC","BTNC","BTS","BTW","BUI","BWG","CBG","CCET","CCN","CCP","CEN","CENTEL","CFRESH","CGD","CGH","CHARAN","CHEWA","CHG","CHO","CHOTI","CHOW","CHUO","CI","CIG","CIMBT","CITY","CK","CKP","CM","CMO","CMR","CNS","CNT","COL","COLOR","COM7","COMAN","CPALL","CPF","CPH","CPI","CPL","CPN","CPR","CRANE","CSC","CSL","CSP","CSR","CSS","CTW","CWT","DAII","DCC","DCON","DCORP","DELTA","DEMCO","DIMET","DNA","DRACO","DRT","DSGT","DTAC","DTC","DTCI","EA","EARTH","EASON","EASTW","ECF","ECL","EE","EFORL","EGCO","EIC","EKH","EMC","EPCO","EPG","ERW","ESSO","ESTAR","EVER","F&D","FANCY","FC","FE","FER","FMT","FN","FNS","FOCUS","FORTH","FPI","FSMART","FSS","FVC","GBX","GC","GCAP","GEL","GENCO","GFPT","GIFT","GJS","GL","GLAND","GLOBAL","GLOW","GOLD","GPSC","GRAMMY","GRAND","GREEN","GSTEL","GTB","GUNKUL","GYT","HANA","HARN","HFT","HMPRO","HOTPOT","HPT","HTC","HTECH","HYDRO","ICC","ICHI","IEC","IFEC","IFS","IHL","ILINK","INET","INOX","INSURE","INTUCH","IRC","IRCP","IRPC","IT","ITD","ITEL","IVL","J","JAS","JCT","JMART","JMT","JSP","JTS","JUBILE","JUTHA","JWD","K","KAMART","KASET","KBANK","KBS","KC","KCAR","KCE","KCM","KDH","KGI","KIAT","KKC","KKP","KOOL","KSL","KTB","KTC","KTIS","KWC","KWG","KYE","L&E","LALIN","LANNA","LDC","LEE","LH","LHBANK","LHK","LIT","LOXLEY","LPH","LPN","LRH","LST","LTX","LVT","M","M-CHAI","MACO","MAJOR","MAKRO","MALEE","MANRIN","MATCH","MATI","MAX","MBAX","MBK","MBKET","MC","MCOT","MCS","MDX","MEGA","METCO","MFC","MFEC","MIDA","MILL","MINT","MJD","MK","ML","MODERN","MONO","MOONG","MPG","MPIC","MSC","MTI","MTLS","NBC","NC","NCH","NCL","NDR","NEP","NETBAY","NEW","NEWS","NINE","NKI","NMG","NNCL","NOBLE","NOK","NPK","NPP","NSI","NTV","NUSA","NWR","NYT","OCC","OCEAN","OGC","OHTL","OISHI","ORI","OTO","PACE","PAE","PAF","PAP","PATO","PB","PCA","PCSGH","PDG","PDI","PE","PERM","PF","PG","PHOL","PICO","PIMO","PJW","PK","PL","PLANB","PLAT","PLE","PM","PMTA","POLAR","POST","PPM","PPP","PPS","PR","PRAKIT","PRANDA","PREB","PRECHA","PRG","PRIN","PRINC","PRO","PSH","PSL","PSTC","PT","PTG","PTL","PTT","PTTEP","PTTGC","PYLON","Q-CON","QH","QLT","QTC","RAM","RATCH","RCI","RCL","RICH","RICHY","RJH","RML","ROBINS","ROCK","ROH","ROJNA","RP","RPC","RS","RWI","S&J","S","S11","SABINA","SALEE","SAM","SAMART","SAMCO","SAMTEL","SANKO","SAPPE","SAT","SAUCE","SAWAD","SAWANG","SC","SCB","SCC","SCCC","SCG","SCI","SCN","SCP","SE-ED","SEAFCO","SEAOIL","SELIC","SENA","SF","SFP","SGF","SGP","SHANG","SIAM","SIM","SIMAT","SINGER","SIRI","SIS","SITHAI","SKR","SLP","SMART","SMIT","SMK","SMM","SMPC","SMT","SNC","SNP","SOLAR","SORKON","SPA","SPACK","SPALI","SPC","SPCG","SPG","SPI","SPORT","SPPT","SPRC","SPVI","SQ","SR","SRICHA","SSC","SSF","SSI","SSSC","SST","STA","STANLY","STAR","STEC","STHAI","STPI","SUC","SUPER","SUSCO","SUTHA","SVH","SVI","SVOA","SWC","SYMC","SYNEX","SYNTEC","T","TACC","TAE","TAKUNI","TAPAC","TASCO","TBSP","TC","TCAP","TCB","TCC","TCCC","TCJ","TCMC","TCOAT","TEAM","TF","TFD","TFG","TFI","TGCI","TGPRO","TH","THAI","THANA","THANI","THCOM","THE","THIP","THL","THRE","THREL","TIC","TICON","TIP","TIPCO","TISCO","TIW","TK","TKN","TKS","TKT","TLUXE","TM","TMB","TMC","TMD","TMI","TMILL","TMT","TMW","TNDT","TNH","TNITY","TNL","TNP","TNPC","TNR","TOG","TOP","TOPP","TPA","TPAC","TPBI","TPC","TPCH","TPCORP","TPIPL","TPOLY","TPP","TR","TRC","TRITN","TRT","TRU","TRUBB","TRUE","TSC","TSE","TSF","TSI","TSR","TSTE","TSTH","TTA","TTCL","TTI","TTL","TTTM","TTW","TU","TUCC","TVD","TVI","TVO","TVT","TWP","TWPC","TWZ","TYCN","U","UAC","UBIS","UEC","UKEM","UMI","UMS","UNIQ","UOBKH","UP","UPA","UPF","UPOIC","UREKA","UT","UTP","UV","UVAN","UWC","VARO","VGI","VIBHA","VIH","VNG","VNT","VPO","VTE","WACOAL","WAVE","WG","WHA","WICE","WIIK","WIN","WINNER","WORK","XO","YCI","YNP","YUASA","ZMICO"];

constructor(public navCtrl: NavController, public navParams: NavParams,private DataServiceProvider:DataServiceProvider) {

}

GetStockList(){

console.log("run === this.DataServiceProvider.PostData({});");

let run = this.DataServiceProvider.GetData({url:"http://localhost:1880/searchstock?name="+this.name}).subscribe(value => {

if (value.access) {

console.log("Send value01", value);

this.stock\_list = value.data;

//return value;

}

else {

console.log("Send value01 else", value);

console.log("error", this.DataServiceProvider.HTTP\_errorhandle(value.error))

//return value;

}

}, error => { console.error(error.message) },

() => { console.log('Completed subscribe007') }

)

}

ViewDetail(page:string,object:any[]){

this.navCtrl.push(page,object);

}

ionViewDidLoad() {

console.log('ionViewDidLoad StockPage');

}

}

Stock.html

<!--

Generated template for the StockPage page.

See http://ionicframework.com/docs/components/#navigation for more info on

Ionic pages and navigation.

-->

<ion-header>

<ion-navbar>

<ion-title>stock</ion-title>

</ion-navbar>

<ion-searchbar

[(ngModel)]="name"

[showCancelButton]="shouldShowCancel"

(ionInput)="GetStockList()"

(ionCancel)="onCancel($event)">

</ion-searchbar>

</ion-header>

<ion-content padding>

<ion-list>

<ion-item-sliding #itemsliding \*ngFor="let stock of stock\_list; let i = index;">

<ion-item (click)="ViewDetail('StockViewPage',stock)">

<h2>{{stock}}</h2>

<button ion-button clear item-end>View</button>

</ion-item>

</ion-item-sliding>

</ion-list>

</ion-content>

Stock-view.ts

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

import { IonicPage, NavController, NavParams, LoadingController } from 'ionic-angular';

import {Chart} from 'Chart.js';

import {DataServiceProvider} from '../../providers/data-service/data-service'

/\*\*

\* Generated class for the StockViewPage page.

\*

\* See https://ionicframework.com/docs/components/#navigation for more info on

\* Ionic pages and navigation.

\*/

@IonicPage()

@Component({

selector: 'page-stock-view',

templateUrl: 'stock-view.html',

})

export class StockViewPage {

name = this.navParams.data;

constructor(public navCtrl: NavController, public navParams: NavParams,private DataServiceProvider:DataServiceProvider,

private loadingCtrl: LoadingController) {

}

ionViewDidLoad() {

console.log('ionViewDidLoad StockViewPage');

this.createlineChart();

this.GetStockInfo();

}

lineChartEl\_AddData(List\_Value: any, List\_Label: any) {

this.linechartValues.push(List\_Value);

this.linechartLabels.push(List\_Label);

this.linechartColours.push(this.getRandomColor());

this.linechartHoverColours.push(this.getRandomColor());

this.lineChartEl.update();

}

ChartRemoveValue() {

let count = this.linechartValues.length;

for (let index = 0; index < count; index++) {

console.log("ChartRemoveValue")

this.linechartValues.shift();

this.linechartLabels.shift();

this.linechartColours.shift();

this.linechartHoverColours.shift();

}

this.lineChartEl.update();

}

getRandomColor() {

var color = Math.floor(0x1000000 \* Math.random()).toString(16);

return '#' + ('000000' + color).slice(-6);

}

GetStockInfo(){

console.log("run === this.DataServiceProvider.PostData({});");

this.presentLoadingCustom();

let run = this.DataServiceProvider.GetData({url:"http://localhost:1880/stocks?name="+this.name}).subscribe(value => {

if (value.access) {

console.log("Send value01", value);

//this.listproduct = value;

this.ChartRemoveValue();

value.data.data.forEach(element => {

this.lineChartEl\_AddData(element.price\_close,element.datetime);

});

this.loadingView.dismiss();

//return value;

}

else {

console.log("Send value01 else", value);

console.log("error", this.DataServiceProvider.HTTP\_errorhandle(value.error))

this.loadingView.dismiss();

//return value;

}

}, error => { console.error(error.message) },

() => { console.log('Completed subscribe007') }

)

}

//strat chart

//lineChart

@ViewChild('linecharthome') linecharthome;

public lineChartEl: any;

linechartLabels:any[] =[];

linechartValues:any[] = [];

linechartColours:any[] = [];

linechartHoverColours:any[] = [];

lineChartadddata(Labels,Values){

this.linechartLabels.push(Labels);

this.linechartValues.push(Values);

this.lineChartEl.update();

}

createlineChart()

{

this.lineChartEl = new Chart(this.linecharthome.nativeElement,

{

type: 'line',

data: {

labels: this.linechartLabels,

datasets: [{

label : this.name,

data : this.linechartValues,

duration : 2000,

easing : 'easeInQuart',

backgroundColor : this.linechartColours,

hoverBackgroundColor : this.linechartHoverColours,

fill          : false

}]

},

options : {

maintainAspectRatio: false,

legend : {

display : true,

boxWidth : 80,

fontSize : 15,

padding : 0

},

scales: {

yAxes: [{

ticks: {

beginAtZero:true,

stepSize: 1,

max : 10

}

}],

xAxes: [{

ticks: {

autoSkip: false

}

}]

}

}

});

}

//end lineChart

loadingView:any;

presentLoadingCustom() {

this.loadingView = this.loadingCtrl.create({

spinner: 'hide',

content: `

<div class="custom-spinner-container">

<div class="custom-spinner-box"><object data='../../assets/icon/loading.svg' type='image/svg+xml'></object></div>

</div>`,

duration: 15000

});

this.loadingView.onDidDismiss(() => {

console.log('Dismissed loading');

});

this.loadingView.present();

}

doRefresh(refresher) {

console.log('Begin async operation', refresher);

////this.GetOrders();

setTimeout(() => {

console.log('Async operation has ended');

refresher.complete();

}, 4000);

}

}

Stock-view.html

<!--

Generated template for the StockViewPage page.

See http://ionicframework.com/docs/components/#navigation for more info on

Ionic pages and navigation.

-->

<ion-header>

<ion-navbar>

<ion-title>History : {{name}}</ion-title>

</ion-navbar>

</ion-header>

<ion-content>

<ion-card>

<ion-item>

<canvas #linecharthome></canvas>

</ion-item>

</ion-card>

<button (click)="GetStockInfo()">dddddddd</button>

</ion-content>

Dataservice provider .ts

import { HttpClient, HttpHeaders } from '@angular/common/http';

import { Injectable } from '@angular/core';

import { Observable } from 'rxjs/Observable';

/\*

Generated class for the DataServiceProvider provider.

See https://angular.io/guide/dependency-injection for more info on providers

and Angular DI.

\*/

@Injectable()

export class DataServiceProvider {

constructor(public HttpClient: HttpClient) {

console.log('Hello DataServiceProvider Provider');

}

public HTTP\_errorhandle(HttpErrorResponse:any){

let error = {title:'',

message:'',

status:''

};

error.title = HttpErrorResponse.name;

error.message = HttpErrorResponse.message;

error.status = HttpErrorResponse.statusText;

return error;

}

///

///

/\* calltest(){

console.log("run === this.DataServiceProvider.PostData({});");

this.presentLoadingCustom();

let run = this.DataServiceProvider.PostData({}).subscribe(value => {

if (value.access) {

console.log("Send value01", value);

this.listproduct = value;

this.loadingView.dismiss();

//return value;

}

else {

console.log("Send value01 else", value);

console.log("error", this.DataServiceProvider.HTTP\_errorhandle(value.error))

this.loadingView.dismiss();

//return value;

}

}, error => { console.error(error.message) },

() => { console.log('Completed subscribe007') }

)

}

loadingView:any;

presentLoadingCustom() {

this.loadingView = this.loadingCtrl.create({

spinner: 'hide',

content: `

<div class="custom-spinner-container">

<div class="custom-spinner-box"><object data='../../assets/icon/loading.svg' type='image/svg+xml'></object></div>

</div>`,

duration: 15000

});

this.loadingView.onDidDismiss(() => {

console.log('Dismissed loading');

});

this.loadingView.present();

}

doRefresh(refresher) {

console.log('Begin async operation', refresher);

////this.GetOrders();

setTimeout(() => {

console.log('Async operation has ended');

refresher.complete();

}, 4000);

} \*/

////

///

public PostData(option: {

url?: any,

token?: any,

body?: any,

timesout?: number,

headers?:HttpHeaders

}): any {

let url = (option.url != null ? option.url : "http://localhost:44300/api/Merchandise/Search");

let body = (option.body != null ? option.body : { "IdentityId": "true" })

let token = (option.token != null ? option.token : "this.AuthServiceProvider.token");

let timeout = (option.timesout > 0 ? option.timesout : 4000);

let headers = (option.headers != null ? option.headers : new HttpHeaders().set('Authorization', token).set('Content-Type', 'application/json'))

let \_Subscribe: any //declaration variable to unsubscribe if any error is occurred

let \_access: any = { access: false, data: [], row: [], md5: [], error: [] }; // value that return to use in another function to determine whether date is complete or not

if (body.IdentityId === null) { //Check data is valid or not

return Observable.throw("Please insert body"); //throw "Please insert body"

}

else {

return Observable.create(observer => {

\_Subscribe = this.HttpClient.post<any>(url, body, {headers: headers}) //Call function PostData from data-service

.subscribe(

data => {

if (data) {

\_access.access = true;

\_access.data = data.data;

\_access.row = data.row;

\_access.md5 = data.md5;

observer.next(\_access);

observer.complete();

}

else {

\_access.access = false;

\_access.error = "data is null";

observer.next(\_access);

observer.error(new Error('PostData Error!'));

observer.complete();

\_Subscribe.unsubscribe();

}

},

err => {

\_access.access = false;

\_access.data = err;

\_access.error = err;

observer.next(\_access);

observer.error(new Error(err));

observer.complete();

\_Subscribe.unsubscribe();

},//can add function to handle the action of error

() => { console.log('PostData Done'); } // when get data is completed. It will console.log('GetMerchandise Done')

);

setTimeout(() => {

//\_access.data = "Timesout";

//\_access.error = "Timesout";

\_Subscribe.unsubscribe();

observer.next(\_access);

observer.complete();

}, timeout);

}, err => console.error('observer : ', err));

}

}

public GetData(option: {

url?: any,

token?: any,

timesout?: number,

headers?:HttpHeaders

}): any {

let url = (option.url != null ? option.url : "http://localhost:44300/api/Merchandise/Search");

let token = (option.token != null ? option.token : "this.AuthServiceProvider.token");

let timeout = (option.timesout > 0 ? option.timesout : 4000);

let headers = (option.headers != null ? option.headers : new HttpHeaders().set('Authorization', token))

let \_Subscribe: any //declaration variable to unsubscribe if any error is occurred

let \_access: any = { access: false, data: [], row: [], md5: [], error: [] }; // value that return to use in another function to determine whether date is complete or not

if (url === null) { //Check data is valid or not

return Observable.throw("Please insert url"); //throw "Please insert body"

}

else {

return Observable.create(observer => {

\_Subscribe = this.HttpClient.get<any>(url, {headers: headers}) //Call function PostData from data-service

.subscribe(

data => {

if (data) {

\_access.access = true;

\_access.data = data;

\_access.row = data.row;

\_access.md5 = data.md5;

observer.next(\_access);

observer.complete();

}

else {

\_access.access = false;

\_access.error = "data is null";

observer.next(\_access);

observer.error(new Error('GetData Error!'));

observer.complete();

\_Subscribe.unsubscribe();

}

},

err => {

\_access.access = false;

\_access.data = err;

\_access.error = err;

observer.next(\_access);

observer.error(new Error(err));

observer.complete();

\_Subscribe.unsubscribe();

},//can add function to handle the action of error

() => { console.log('GetData Done'); } // when get data is completed. It will console.log('GetMerchandise Done')

);

setTimeout(() => {

//\_access.data = "Timesout";

//\_access.error = "Timesout";

\_Subscribe.unsubscribe();

observer.next(\_access);

observer.complete();

}, timeout);

}, err => console.error('observer : ', err));

}

}

/////distinct

public distinct\_utc(\_ListOfObject) { // utc time distinct

let \_sum = [];

var mapfoo = \_ListOfObject.map(ListOfObject => {

let index = \_sum.findIndex(sum => {

let sum\_time = new Date(sum.deliveryTimes);

let ListOfObject\_time = new Date(ListOfObject.deliveryTimes);

return (sum\_time.getDay() == ListOfObject\_time.getDay() &&

sum\_time.getMonth() == ListOfObject\_time.getMonth() &&

sum\_time.getFullYear() == ListOfObject\_time.getFullYear());

});

if (index >= 0) { // if in list do something

//\_sum[index].qty += ListOfObject.qty;

\_sum[index].count += 1;

}

else {

\_sum.push({ deliveryTimes: ListOfObject.deliveryTimes, count: 1 })// if not in list do some thing

}

})

\_sum = \_sum.sort((a, b) => a.week - b.week);

console.log(\_sum)//{week: 1, qty: 2, count: 2}, {week: 2, qty: 4, count: 2}

return \_sum;

}

//Search

//filtered:any =[];

//varfilterall:any = {order:'',filter:'',range:''};

public filterall(ListOfObject,filter:{order?:any,filter?:any,range?:any,type?:any}){

console.log("run","filterall()")

let filtered: any;

console.log(filter);

filtered = ListOfObject;

/\* let start;

let end;

let now = new Date();

switch (this.varfilterall.range) {

case 'today':

start = new Date(now.setHours(0, 0, 0, 0));

end = new Date(now.setHours(23, 59, 59, 999));

break;

case 'week':

start = new Date(now.setHours(0, 0, 0, 0));

end = new Date(

now.getFullYear(),

now.getMonth(),

now.getDate() + (7 - now.getDay()), 23, 59, 59, 999

);

break;

case 'all':

start = new Date(

now.getFullYear()-(1000),

now.getMonth(),

now.getDate() + (7 - now.getDay()), 23, 59, 59, 999

);

end = new Date(

now.getFullYear()+(1000),

now.getMonth(),

now.getDate() + (7 - now.getDay()), 23, 59, 59, 999

);

break;

default:

start = new Date(now.setHours(0, 0, 0, 0));

end = new Date(now.setHours(23, 59, 59, 999));

break;

// range = this.filters.customRange;

// start = range[0];

// end = range[1];

} \*/

if(filter.filter != ''){

filtered = filtered.filter(o => o.name.includes(filter.filter));

console.log("filter");

}

if(filter.type != ''){

filtered = filtered.filter(o => o.type == filter.type);

console.log("Type");

}

if(filter.range != ''){

filtered = filtered.filter(o => o.range == filter.range);

console.log("range");

}

/\* if(this.varfilterall.range != ''){

this.filtered = this.filtered.filter(o => Date.parse(o.orderTimes) >= start && Date.parse(o.orderTimes) <= end);

console.log("date filter");

} \*/

/\* if(this.varfilterall.orderIsCanceled == 'true'){

this.filtered = this.filtered.filter(o => o.orderIsCanceled == true);

console.log("date orderIsCanceled");

} \*/

/\* if(this.varfilterall.order =='Ascending'){

this.filtered.sort((n1,n2) => n1.merchandiseQTY - n2.merchandiseQTY)

console.log("2");

}

else if(this.varfilterall.order == 'Descending'){

this.filtered.sort((n1,n2) => n1.merchandiseQTY - n2.merchandiseQTY).reverse();

console.log("3");

} \*/

console.log("varfilterall", filtered );

return filtered;

}

//end Search

public filterall\_Sale(ListOfObject,filter:{order?:any,filter?:any,range?:any,type?:any}){

console.log("run","filterall()")

let filtered: any;

console.log(filter);

filtered = ListOfObject;

/\* let start;

let end;

let now = new Date();

switch (this.varfilterall.range) {

case 'today':

start = new Date(now.setHours(0, 0, 0, 0));

end = new Date(now.setHours(23, 59, 59, 999));

break;

case 'week':

start = new Date(now.setHours(0, 0, 0, 0));

end = new Date(

now.getFullYear(),

now.getMonth(),

now.getDate() + (7 - now.getDay()), 23, 59, 59, 999

);

break;

case 'all':

start = new Date(

now.getFullYear()-(1000),

now.getMonth(),

now.getDate() + (7 - now.getDay()), 23, 59, 59, 999

);

end = new Date(

now.getFullYear()+(1000),

now.getMonth(),

now.getDate() + (7 - now.getDay()), 23, 59, 59, 999

);

break;

default:

start = new Date(now.setHours(0, 0, 0, 0));

end = new Date(now.setHours(23, 59, 59, 999));

break;

// range = this.filters.customRange;

// start = range[0];

// end = range[1];

} \*/

if(filter.filter != ''){

filtered = filtered.filter(o => o.name.includes(filter.filter));

console.log("filter");

}

if(filter.type != ''){

filtered = filtered.filter(o => o.type == filter.type);

console.log("Type");

}

if(filter.range != ''){

filtered = filtered.filter(o => o.range == filter.range);

console.log("range");

}

/\* if(this.varfilterall.range != ''){

this.filtered = this.filtered.filter(o => Date.parse(o.orderTimes) >= start && Date.parse(o.orderTimes) <= end);

console.log("date filter");

} \*/

/\* if(this.varfilterall.orderIsCanceled == 'true'){

this.filtered = this.filtered.filter(o => o.orderIsCanceled == true);

console.log("date orderIsCanceled");

} \*/

/\* if(this.varfilterall.order =='Ascending'){

this.filtered.sort((n1,n2) => n1.merchandiseQTY - n2.merchandiseQTY)

console.log("2");

}

else if(this.varfilterall.order == 'Descending'){

this.filtered.sort((n1,n2) => n1.merchandiseQTY - n2.merchandiseQTY).reverse();

console.log("3");

} \*/

console.log("varfilterall", filtered );

return filtered;

}

}