**Documentation of Angular**

**Stapes**

1. **Open new Angular project With .netcore with angular**
2. **Create a database(Db Model. ConnectionString. Configuration and View Model)**
3. **it is a one to many Db Model**
4. **install (EFCOre 3.1.18 Sql 3.1.18 EF Tools)**
5. **Take a API controller.**
6. **Go to Your ClientApp Folder And Delete It**
7. **In this Folder go to cmd And Follow blew Command**
8. **ng new ClientApp –skip-tests**
9. **cd ClientApp**
10. **ng add @angular/material**
11. **dotnet build**
12. **ng g c components/home**
13. **ng g m modules/common/mat**
14. **ng g c components/**
15. **go to module and arrange between app.module and mat.module. in your need**
16. **in app.module....**

  imports: [

  BrowserModule,

  AppRoutingModule,

  BrowserAnimationsModule,

  LayoutModule,

  HttpClientModule,

  FormsModule,

  MatModule

],

1. **in mat.module**

const modules = [

  CommonModule,

  LayoutModule,

  MatToolbarModule,

  MatButtonModule,

  MatSidenavModule,

  MatIconModule,

  MatListModule,

];

**ng g @angular/material:navigation -------/-----**

1. **Go to project and arrange app.component.html and main-nav**
2. **writh nav.component.css**
3. **spacify rout in your need in app.routing.module**
4. **depend on route spacify routerlink in nav.component.html like..**

<mat-nav-list>

      <a mat-list-item [routerLink]="['']"><mat-icon style="margin-right:10px">home</mat-icon> Home </a>

      <a mat-list-item [routerLink]="['brands']"><mat-icon  style="margin-right:10px">bookmarks</mat-icon> Brands  </a>

      <a mat-list-item [routerLink]="['products']"><mat-icon style="margin-right:10px">category</mat-icon>Products </a>

    </mat-nav-list>

1. **nav.component.css take its...**

.sidenav-container {

  height: 100%;

}

.sidenav {

  width: 260px;

}

.sidenav .mat-toolbar {

  background: inherit;

}

.mat-toolbar.mat-primary {

  position: sticky;

  top: 0;

  z-index: 1;

}

.filler{

    flex:1 1 auto;

}

1. ng g cl modules/Product
2. ng g cl modules/Brand
3. ng g cl modules/Product-input
4. ng g cl modules/Product-edit
5. ng g cl modules/Product
6. ng g cl modules/brand-
7. add in brand.tc

export class Brand {

  constructor(

    public brandId?: number,

    public brandName?: string,

    public origin?: string

  ) { }

}

1. add in product.ts.....

export class Product {

  constructor(

    public productId?: number,

    public productName?: string,

    public price?: number,

    public releaseDate?: Date,

    public picture?: string,

    public discontinued?: boolean,

    public brandId?: number

  ) { }

}

1. ng g c components/brand/ brand -view
2. ng g c components/brand/ brand -edit
3. ng g c components/brand/ brand -create
4. ng g c components/product/product-view
5. ng g c components/product/product-create
6. ng g c components/product/product-edit

1. **add in app.rote.module**

const routes: Routes = [

  { path: '', component: HomeComponent },

  { path: 'home', component: HomeComponent },

  { path: 'brands', component: BrandViewComponent },

  { path: 'brand-add', component: BrandCreateComponent },

  { path: 'brand-edit/:id', component: BrandEditComponent },

  { path: 'products', component: ProductViewComponent },

  { path: 'product-add', component: ProductCreateComponent },

  { path: 'product-edit/:id', component: ProductEditComponent }

];

1. **give routerlink in nav.html**
2. **ng g s services/data**
3. **ng g s services/notify**
4. **give service in app.module provider with entrypoint...**

 entryComponents: [ConfirmDialogComponent],

  providers: [DataService, NotifyService],

  bootstrap: [AppComponent]

})

1. **give your need module in mat.module like..**

const modules = [

  CommonModule,

  LayoutModule,

  MatToolbarModule,

  MatButtonModule,

  MatSidenavModule,

  MatIconModule,

  MatListModule,

  MatTableModule,

  MatPaginatorModule,

  MatSortModule,

  MatCardModule,

  MatFormFieldModule,

  MatInputModule,

  MatSelectModule,

  MatProgressBarModule,

  MatSnackBarModule,

  MatDialogModule,

  MatCheckboxModule

];

1. **add in dataservice**

export class DataService {

  constructor(private http: HttpClient) { }

  getBrand(): Observable<Brand[]> {

    return this.http.get<Brand[]>('/api/Brands');

  }

  getBrandById(id: number): Observable<Brand> {

    return this.http.get<Brand>(`/api/Brands/${id}`);

  }

  postBrand(data: Brand): Observable<Brand> {

    return this.http.post<Brand>(`/api/Brands`, data);

  }

  putBrand(data: Brand): Observable<any> {

    return this.http.put<any>(`/api/Brands/${data.brandId}`, data);

  }

  deleteBrand(id: number): Observable<Brand> {

    return this.http.delete<Brand>(`/api/Brands/${id}`);

  }

  /////////////////////////////////////////////////////

  getProduct(): Observable<Product[]> {

    return this.http.get<Product[]>('/api/Products');

  }

  getProductById(id: number): Observable<Product> {

    return this.http.get<Product>(`/api/Products/${id}`);

  }

  postProduct(data: Product): Observable<Product> {

    return this.http.post<Product>(`/api/Products`, data);

  }

  putProduct(data: Product): Observable<any> {

    return this.http.put<any>(`/api/Products/${data.productId}`, data);

  }

  deleteProduct(id: number): Observable<Product> {

    return this.http.delete<Product>(`/api/Products/${id}`);

  }

}

1. **in brand-view.compnents.ts..**

export class BrandViewComponent implements OnInit {

  brands: Brand[] = [];

  dataSource: MatTableDataSource<Brand> = new MatTableDataSource(this.brands); /// Initiates from view

  @ViewChild(MatPaginator, { static: false }) paginator!: MatPaginator;

  @ViewChild(MatSort, { static: false }) sort!: MatSort;

  columnList = ['name', "origin", "actions"];

  constructor(

    private dataSvc: DataService,

    private notifySvc: NotifyService,

    private delDiag: MatDialog

  ) { }

  //confirmDelete(brand: Brand): void {

  //  this.delDiag.open(ConfirmDialogComponent, {

  //    width: '400px'

  //  }).afterClosed()

  //    .subscribe(r => {

  //      if (r) {

  //        this.dataSvc.deleteBrand(Number(brand.brandId))

  //          .subscribe(r => {

  //            this.dataSource.data = this.dataSource.data.filter(x => x.brandId != brand.brandId);

  //          }, err => {

  //            this.notifySvc.fail("Failed to delete", "DISMISS")

  //          });

  //      }

  //      else {

  //      }

  //    });

  //}

  ngOnInit(): void {

    this.dataSvc.getBrand()

      .subscribe(r => {

        this.brands = r;

        this.dataSource.data = this.brands;

        this.dataSource.paginator = this.paginator;

        this.dataSource.sort = this.sort;

      }, err => {

      })

  }

}

1. **in brand-view.components.html**

<mat-card class="card-container">

<mat-card-header>

  <mat-card-title class="card-title">

    Products

  </mat-card-title>

  <mat-card-subtitle class="card-subtitle">

    Product List

  </mat-card-subtitle>

  <span class="filler"></span>

  <a [routerLink]="['/brand-add']" mat-mini-fab color="primary"><mat-icon>add</mat-icon></a>

</mat-card-header>

<mat-card-content>

  <div class="table-container">

    <mat-table [dataSource]="dataSource" matSort>

      <ng-container matColumnDef="name">

        <mat-header-cell \*matHeaderCellDef mat-sort-header>Brand Name</mat-header-cell>

        <mat-cell \*matCellDef="let element">**{{**element.brandName**}}**</mat-cell>

      </ng-container>

      <ng-container matColumnDef="origin">

        <mat-header-cell \*matHeaderCellDef mat-sort-header>Origin</mat-header-cell>

        <mat-cell \*matCellDef="let element">**{{**element.origin **}}**</mat-cell>

      </ng-container>

      <ng-container matColumnDef="actions">

        <mat-header-cell \*matHeaderCellDef>...</mat-header-cell>

        <mat-cell \*matCellDef="let element">

          <button mat-icon-button [routerLink]="['/brand-edit', element.brandId]"><mat-icon>edit</mat-icon></button>

          //<button mat-icon-//button color="warn" (click)="confirmDelete(element)"><mat-icon>delete</mat-//icon></button>

        </mat-cell>

      </ng-container>

      <mat-header-row \*matHeaderRowDef="columnList"></mat-header-row>

      <mat-row \*matRowDef="let row; columns: columnList;"></mat-row>

    </mat-table>

    <mat-paginator [pageSizeOptions]="[5, 10, 25, 100]" [pageSize]="5" showFirstLastButtons></mat-paginator>

  </div>

</mat-card-content>

</mat-card>

1. **in brand-view.components.css**

.card-container{

    box-sizing:border-box;

    width:calc(100% - 30px);

    margin:25px auto 10px auto;

}

.table-container {

  box-sizing: border-box;

  width: calc(100% - 20px);

  margin: 0 auto;

}

1. add MatCardModule in MatModule....
2. in style.css..

.top-container{

    box-sizing:border-box;

    width:98%;

    margin:0 auto;

}

.filler {

    flex: 1 1 auto;

}

1. for Form add in app.module FormsModule
2. in brand-create.components.html..

<div class="form-container mat-typography">

  <form #addBrand="ngForm" (ngSubmit)="insert(addBrand)" class="form-container">

    <h3>Create new product</h3>

    <mat-form-field class="f-w">

      <mat-label>Product name</mat-label>

      <input matInput name="name" id="name" [(ngModel)]="brand.brandName" required #name="ngModel">

      <mat-error \*ngIf="name.pristine && name.invalid">Name required</mat-error>

    </mat-form-field>

    <mat-form-field class="f-w">

      <mat-label>Country of origin</mat-label>

      <input matInput name="origin" id="origin" [(ngModel)]="brand.origin" required #origin="ngModel">

      <mat-error \*ngIf="origin.touched && origin.invalid">Origin required</mat-error>

    </mat-form-field>

    <div class="f-w">

      <button [disabled]="addBrand.invalid" mat-raised-button type="submit" color="accent"><mat-icon>save\_alt</mat-icon> Save</button>

    </div>

    <div class="f-w">

      <a mat-flat-button [routerLink]="['/brands']">Back to list</a>

    </div>

  </form>

</div>

1. in brand-create.components.ts..

@Component({

  selector: 'app-brand-create',

  templateUrl: './brand-create.component.html',

  styleUrls: ['./brand-create.component.css']

})

export class BrandCreateComponent implements OnInit {

  brand: Brand = new Brand();

  constructor(

    private dataSvc: DataService,

    private notifySvc: NotifyService

  ) { }

  insert(f: NgForm): void {

    if (f.invalid) return;

    this.dataSvc.postBrand(this.brand)

      .subscribe(r => {

        this.notifySvc.success("Data saved successfully", "DISMISS")

      }, err => {

        this.notifySvc.fail("Failed to save data", "DISMISS")

      })

  }

  ngOnInit(): void {

  }

}

1. in brand-creatte.components.css

.form-container

{

    width:65%;

    margin: 5px auto 10px auto;

}

.f-w{

    width:100%

}

1. **in notify.services..**

export class NotifyService {

  constructor(private snackBar: MatSnackBar) { }

  success(message: string, action: string): void {

    let config: MatSnackBarConfig = {

      duration: 3000,

      panelClass: 'primary'

      //duration: 3000,

      //horizontalPosition: 'right',

      //verticalPosition: 'top'

    }

    this.snackBar.open(message, action, config);

  }

  fail(message: string, action: string): void {

    let config: MatSnackBarConfig = {

      duration: 3000,

      panelClass: 'warb'

      //duration: 3000,

      //horizontalPosition: 'right',

      //verticalPosition: 'top'

    }

    this.snackBar.open(message, action, config);

  }

}

1. in brand-edit.components.html

<div class="form-container mat-typography">

  <form #editBrand="ngForm" (ngSubmit)="update(editBrand)" class="form-container">

    <h3>Create new product</h3>

    <mat-form-field class="f-w">

      <mat-label>Product name</mat-label>

      <input matInput name="name" id="name" [(ngModel)]="brand.brandName" required #name="ngModel">

      <mat-error \*ngIf="name.pristine && name.invalid">Name required</mat-error>

    </mat-form-field>

    <mat-form-field class="f-w">

      <mat-label>Country of origin</mat-label>

      <input matInput name="origin" id="origin" [(ngModel)]="brand.origin" required #origin="ngModel">

      <mat-error \*ngIf="origin.touched && origin.invalid">Origin required</mat-error>

    </mat-form-field>

    <div class="f-w">

      <button [disabled]="editBrand.invalid" mat-raised-button type="submit" color="accent"><mat-icon>save\_alt</mat-icon> Save</button>

    </div>

    <div class="f-w">

      <a mat-flat-button [routerLink]="['/brands']">Back to list</a>

    </div>

  </form>

</div>

1. in brand-edit.components.ts

export class BrandEditComponent implements OnInit {

  brand: Brand = new Brand();

  constructor(

    private dataSvc: DataService,

    private notifySvc: NotifyService,

    private activateRoute: ActivatedRoute

  ) { }

  update(f: NgForm): void {

    if (f.invalid) return;

    console.log(this.brand);

    this.dataSvc.putBrand(this.brand)

      .subscribe(r => {

        this.notifySvc.success("Data update successfully", "DISMISS")

      }, err => {

        this.notifySvc.fail("Failed to update data", "DISMISS")

      });

  }

  ngOnInit(): void {

    let id = this.activateRoute.snapshot.params.id

    this.dataSvc.getBrandById(id)

      .subscribe(r => {

        this.brand = r;

      }, err => {

        this.notifySvc.fail("Failed to fetch data from server", "DISMISS")

      })

    console.log(id);

  }

}

1. in brand-edit.components.css

.form-container {

  width: 65%;

  margin: 5px auto 10px auto;

}

.f-w {

  width: 100%

}

1. ng g c components/confirm-dialog
2. add entryComponents in app.module

entryComponents: [ConfirmDialogComponent],

and in @ngModule in app.module ..

ConfirmDialogComponent

1. in confirm-dialog.html

<div mat-dialog-title color="warn">

  Confirm deletion?

</div>

<mat-dialog-content>

  <strong>Are you sure to delete this item?</strong>

</mat-dialog-content>

<mat-dialog-actions align="end">

  <button mat-button [mat-dialog-close]="false" cdkFocusInitial>Cancel</button>

  <button mat-raised-button color="warn" [mat-dialog-close]="true">Yes</button>

</mat-dialog-actions>

1. in confirm-dialog.ts
2. add in brand-view.components.html

<button mat-icon-button color="warn" (click)="confirmDelete(element)"><mat-icon>delete</mat-icon></button>

1. add in brand-view.components.ts

constructor(

    private dataSvc: DataService,

    private notifySvc: NotifyService,

    private delDiag: MatDialog

  ) { }

  confirmDelete(brand: Brand): void {

    this.delDiag.open(ConfirmDialogComponent, {

      width: '400px'

    }).afterClosed()

      .subscribe(r => {

        if (r) {

          this.dataSvc.deleteBrand(Number(brand.brandId))

            .subscribe(r => {

              this.dataSource.data = this.dataSource.data.filter(x => x.brandId != brand.brandId);

            }, err => {

              this.notifySvc.fail("Failed to delete", "DISMISS")

            });

        }

        else {

        }

For product Model

1. add in product-input model

export class ProductVM {

  constructor(

    public productId?: number,

    public productName?: string,

    public price?: number,

    public releaseDate?: Date,

    public picture?: File,

    public discontinued?: boolean,

    public brandId?: number

  ) { }

}

1. in product api controller for upload

[HttpPost("Upload")]

       public async Task<ActionResult<Product>> UploadProduct(ProductInputModel m)

       {

           if (ModelState.IsValid)

           {

               var p = new Product

               {

                   ProductName =m.ProductName,

                   Price =m.Price,

                   ReleaseDate = m.ReleaseDate,

                   Discontinued = m.Discontinued,

                   BrandId = m.BrandId,

                   Picture = "no-pic.png"

               };

               if(m.Picture!= null && m.Picture.Length> 0)

               {

                   string filePath = Path.Combine(env.WebRootPath, "Uploads");

                   string fileName = Guid.NewGuid() + Path.GetExtension(m.Picture.FileName);

                   FileStream fs = new FileStream(Path.Combine(filePath, fileName), FileMode.Create);

                   await m.Picture.CopyToAsync(fs);

                   p.Picture = fileName;

               }

               \_context.Products.Add(p);

               await \_context.SaveChangesAsync();

               return p;

           }

           return Ok();

       }

1. in product-view.components.ts

  constructor(

    private dataSvc: DataService,

    private notifySvc: NotifyService,

    private delDiag: MatDialog

  ) { }

  ngOnInit(): void {

    this.dataSvc.getProduct()

      .subscribe(r => {

        this.products = r;

        console.log(this.products);

        this.dataSource.data = this.products;

        this.dataSource.paginator = this.paginator;

        this.dataSource.sort = this.sort;

      }, err => {

        this.notifySvc.success("Failed to load data from server.", "DISMISS");

      })

  }}

1. in product-view.components.html

<mat-card class="card-container">

  <mat-card-header>

    <mat-card-title class="card-title">

      Products

    </mat-card-title>

    <mat-card-subtitle class="card-subtitle">

      Product List

    </mat-card-subtitle>

    <span class="filler"></span>

    <a [routerLink]="['/product-add']" mat-mini-fab color="primary"><mat-icon>add</mat-icon></a>

  </mat-card-header>

  <mat-card-content>

    <div class="table-container">

      <mat-table [dataSource]="dataSource" matSort>

        <ng-container matColumnDef="picture">

          <mat-header-cell \*matHeaderCellDef mat-sort-header>Product Name</mat-header-cell>

          <mat-cell \*matCellDef="let element">

            <img class="row-image" src="/Uploads/**{{**element.picture**}}**" />

          </mat-cell>

        </ng-container>

        <ng-container matColumnDef="name">

          <mat-header-cell \*matHeaderCellDef mat-sort-header>Product Name</mat-header-cell>

          <mat-cell \*matCellDef="let element">**{{**element.productName**}}**</mat-cell>

        </ng-container>

        <ng-container matColumnDef="price">

          <mat-header-cell \*matHeaderCellDef mat-sort-header>Origin</mat-header-cell>

          <mat-cell \*matCellDef="let element">**{{**element.price **}}**</mat-cell>

        </ng-container>

        <ng-container matColumnDef="release">

          <mat-header-cell \*matHeaderCellDef mat-sort-header>Release Date</mat-header-cell>

          <mat-cell \*matCellDef="let element">**{{**element.releaseDate | date:'yyyy-MM-dd' **}}**</mat-cell>

        </ng-container>

        <ng-container matColumnDef="discontinued">

          <mat-header-cell \*matHeaderCellDef mat-sort-header>Discontinued?</mat-header-cell>

          <mat-cell \*matCellDef="let element">**{{**element.discontinued ? "Yes": "No"**}}**</mat-cell>

        </ng-container>

        <ng-container matColumnDef="brand">

          <mat-header-cell \*matHeaderCellDef mat-sort-header>Brand</mat-header-cell>

          <mat-cell \*matCellDef="let element">**{{**element.brandId **}}**</mat-cell>

        </ng-container>

        <ng-container matColumnDef="actions">

          <mat-header-cell \*matHeaderCellDef>...</mat-header-cell>

          <mat-cell \*matCellDef="let element">

            <button mat-icon-button [routerLink]="['/brand-edit', element.brandId]"><mat-icon>edit</mat-icon></button>

            <button mat-icon-button color="warn"><mat-icon>delete</mat-icon></button>

          </mat-cell>

        </ng-container>

        <mat-header-row \*matHeaderRowDef="columnList"></mat-header-row>

        <mat-row \*matRowDef="let row; columns: columnList;"></mat-row>

      </mat-table>

      <mat-paginator [pageSizeOptions]="[5, 10, 25, 100]" [pageSize]="5" showFirstLastButtons></mat-paginator>

    </div>

  </mat-card-content>

</mat-card>

1. in product-view.components.css

.card-container {

  box-sizing: border-box;

  width: calc(100% - 30px);

  margin: 25px auto 10px auto;

}

.table-container {

  box-sizing: border-box;

  width: calc(100% - 20px);

  margin: 0 auto;

}

.row-image {

  max-height: 40px;

  max-width: 40px;

  overflow: hidden;

  border: solid 1px #FFFFFF;

  -webkit-border-radius: 50%;

  border-radius: 50%;

  -webkit-transition: -webkit-box-shadow 0.3s ease;

  transition: box-shadow 0.3s ease;

  /\*-webkit-box-shadow: 0px 0px 0px 8px rgba(0, 0, 0, 0.06);

  box-shadow: 0px 0px 0px 8px rgba(0, 0, 0, 0.06);\*/

}

1. in product-create.components.html

1. in product-create.components.ts

**It is a reactive form and in angular we cannot uploade file in text.so at first we insert data than we uploade image. thuse here have two part**

**for inserting data .one for data another for image uploade.**

selectedFile!: File;

  product: Product = new Product();

  brands: Brand[] = [];

//for initiate formfield in formGroup

  productForm: FormGroup = new FormGroup({

    productName: new FormControl('', Validators.required),

    price: new FormControl('', Validators.required),

    releaseDate: new FormControl('', Validators.required),

    picture: new FormControl('', Validators.required),

    discontinued: new FormControl(''),

    brandId: new FormControl('', Validators.required),

  });

  constructor(

    private dataSvc: DataService,

    private notifyService: NotifyService,

    private datePipe: DatePipe

  ) { }

//for get file and send to formControll

  get f() {

    return this.productForm.controls;

  }

//for Changing file

  onChange(event: any): void {

    //console.log(event.target.files[0]);

    this.selectedFile = event.target.files[0];

    console.log(this.selectedFile != null)

  }

//for inserting data to database

  insert(): void {

    if (this.productForm.invalid) return;

    console.log(this.productForm.value);

    Object.assign(this.product, this.productForm.value);

    let data = {

      productName: this.product.productName,

      price: this.product.price,

      releaseDate: this.datePipe.transform(this.product.releaseDate, 'yyyy-MM-dd'),

      discontinued: this.product.discontinued == undefined ? false : true,

      brandId: this.product.brandId,

      picture: 'no-pic.png'

    };

    this.dataSvc.postProduct(data)

      .subscribe(r => {

        this.uploadImage(Number(r.productId))

      }, err => {

        this.notifyService.fail("Failed to save product", 'DISMISS')

      })

  }

//for updload image file

  uploadImage(id: number): void {

    const reader = new FileReader();

    reader.addEventListener('load', (event: any) => {

      this.dataSvc.uploadImage(id, this.selectedFile).subscribe(

        (res) => {

          this.notifyService.success("Data saved", 'DISMISS')

          this.productForm.reset();

        },

        (err) => {

          this.notifyService.fail("Failed to save image. Please update it in edit page", 'DISMISS')

        });

    });

    reader.readAsDataURL(this.selectedFile);

  }

  ngOnInit(): void {

    this.dataSvc.getBrand()

      .subscribe(x => { this.brands = x; }, err => {

      });

  }

1. in data service write for uploade image..

uploadImage(id: number, f: File): Observable<ImagePathResponse> {

    const formData = new FormData();

    formData.append('file', f);

    return this.http.post<ImagePathResponse>(`/api/Products/Upload/${id}`, formData);

  }

1. **in product-create.components.html**

<form class="form-container" [formGroup]="productForm" (ngSubmit)="insert()">

  <div class="f-w">

    <h3>Create new product</h3>

  </div>

  <mat-form-field class="f-w">

    <mat-label>Product name</mat-label>

    <input matInput formControlName="productName">

    <mat-error \*ngIf="f.productName.touched && f.productName.errors">Product name required</mat-error>

  </mat-form-field>

  <mat-form-field class="f-w">

    <mat-label>Price</mat-label>

    <input matInput formControlName="price">

    <mat-error \*ngIf="f.price.touched && f.price.errors">Price required</mat-error>

  </mat-form-field> <mat-form-field class="f-w">

    <mat-label>Release date</mat-label>

    <input matInput formControlName="releaseDate" [matDatepicker]="picker">

    <mat-datepicker-toggle matSuffix [for]="picker"></mat-datepicker-toggle>

    <mat-datepicker #picker></mat-datepicker>

    <mat-error \*ngIf="f.releaseDate.touched && f.releaseDate.errors">Release date required</mat-error>

  </mat-form-field>

  <mat-form-field class="f-w">

    <mat-label>Brand</mat-label>

    <mat-select formControlName="brandId">

      <mat-option \*ngFor="let b of brands" [value]="b.brandId">

**{{**b.brandName**}}**

      </mat-option>

    </mat-select>

    <mat-error \*ngIf="f.brandId.touched && f.brandId.errors">Brand required</mat-error>

  </mat-form-field>

  <mat-form-field appearance="standard" class="f-w">

    <mat-label>Product ficture</mat-label>

    <ngx-mat-file-input formControlName="picture" [accept]="'image/\*'" (change)="onChange($event)"></ngx-mat-file-input>

    <mat-icon matSuffix>insert\_photo</mat-icon>

    <mat-error \*ngIf="f.picture.touched && f.picture.errors">Picture required</mat-error>

  </mat-form-field>

  <div class="f-w">

    <mat-label>Discontinued</mat-label>

    <mat-checkbox formControlName="discontinued" style="margin-left:15px;">

    </mat-checkbox>

    <mat-error \*ngIf="f.discontinued.touched && f.discontinued.errors">Price required</mat-error>

  </div>

  <div class="f-w">

    <button [disabled]="productForm.invalid" mat-raised-button type="submit" color="accent"><mat-icon>save\_alt</mat-icon> Save</button>

  </div>

  <div class="f-w">

    <a mat-flat-button [routerLink]="['/products']">Back to list</a>

  </div>

</form>

1. in product-create.components.css

.form-container {

  width: 65%;

  margin: 5px auto 10px auto;

}

.f-w {

  width: 100%;

  margin-bottom: .5rem;

}

1. **in edit-product.components.ts**

**It is a reactive form and in angular we cannot uploade file in text.so at first we insert data than we uploade image. thuse here have two part**

**for inserting data .one for data another for image uploade.**

selectedFile!: File;

  product!: Product;

  brands: Brand[] = [];

//for initiate filede to formgroup

  productForm: FormGroup = new FormGroup({

    productName: new FormControl('', Validators.required),

    price: new FormControl('', Validators.required),

    releaseDate: new FormControl('', Validators.required),

    picture: new FormControl(''),

    discontinued: new FormControl(''),

    brandId: new FormControl('', Validators.required),

  });

  constructor(

    private dataSvc: DataService,

    private notifyService: NotifyService,

    private datePipe: DatePipe,

    private activatedRoute: ActivatedRoute

  ) { }

//for update data

  update(): void {

    if (this.productForm.invalid) return;

    let data = new ProductUpdate();

    data.productId = this.product.productId;

    data.productName = this.f.productName.value;

    data.price = this.f.price.value;

//for tracsfer date

    let dt = this.datePipe.transform(this.f.releaseDate.value, 'yyyy-MM-dd')

    data.releaseDate =  dt== null ? "" : dt;

    data.brandId = this.f.brandId.value;

    data.discontinued = this.f.discontinued.value == "" ? false : true;

    data.picture = this.product.picture;

    //this.dataSvc.putBrand(this.product)

    this.dataSvc.putProduct(data)

      .subscribe(

        r => {

          if (this.selectedFile != null && this.product.productId) {

            this.uploadImage(this.product.productId);

          }

          else {

            this.notifyService.success("Data updated", "DISMISS");

            this.product.productName = this.f.productName.value;

            this.product.price = this.f.price.value;

            this.product.releaseDate = this.f.releaseDate.value;

            this.product.brandId = this.f.brandId.value;

            this.product.discontinued = this.f.discontinued.value == "" ? false : true;

          }

        },

        err => {

          this.notifyService.fail("Data update failed", "DISMISS");

        }

      )

  }

//for image uploade

  uploadImage(id: number): void {

    const reader = new FileReader();

    reader.addEventListener('load', (event: any) => {

      this.dataSvc.uploadImage(id, this.selectedFile).subscribe(

        (res) => {

          this.notifyService.success("Data saved", 'DISMISS')

          this.notifyService.success("Data updated", "DISMISS");

          this.product.productName = this.f.productName.value;

          this.product.price = this.f.price.value;

          this.product.releaseDate = this.f.releaseDate.value;

          this.product.brandId = this.f.brandId.value;

          this.product.discontinued = this.f.discontinued.value == "" ? false : true;

          this.product.picture = res.imagePath;

        },

        (err) => {

          this.notifyService.fail("Failed to save image. Please update it in edit page", 'DISMISS')

        });

    });

    reader.readAsDataURL(this.selectedFile);

  }

  get f() {

    return this.productForm.controls;

  }

//for file change from form

  onChange(event: any): void {

    //console.log(event.target.files[0]);

    this.selectedFile = event.target.files[0];

    console.log(this.selectedFile != null)

  }

  ngOnInit(): void {

//get product data by id which we click

    let id: number = this.activatedRoute.snapshot.params.id;

    this.dataSvc.getProductById(id)

      .subscribe(r => {

        this.product = r;

        this.f.productName.setValue(this.product.productName);

        this.f.price.setValue(this.product.price);

        this.f.releaseDate.setValue(this.product.releaseDate);

        this.f.brandId.setValue(this.product.brandId);

        this.f.discontinued.setValue(this.product.discontinued);

      }, err => {

        this.notifyService.fail("Faile to load product", "DISMISS");

      });

    this.dataSvc.getBrand()

      .subscribe(x => { this.brands = x; }, err => {

      });

  }

1. i edit-product.components.html..

<form class="form-container" [formGroup]="productForm" (ngSubmit)="update()">

  <div class="f-w">

    <h3>Edit product</h3>

  </div>

  <mat-form-field class="f-w">

    <mat-label>Product name</mat-label>

    <input matInput formControlName="productName">

    <mat-error \*ngIf="f.productName.touched && f.productName.errors">Product name required</mat-error>

  </mat-form-field>

  <mat-form-field class="f-w">

    <mat-label>Price</mat-label>

    <input matInput formControlName="price">

    <mat-error \*ngIf="f.price.touched && f.price.errors">Price required</mat-error>

  </mat-form-field> <mat-form-field class="f-w">

    <mat-label>Release date</mat-label>

    <input matInput formControlName="releaseDate" [matDatepicker]="picker">

    <mat-datepicker-toggle matSuffix [for]="picker"></mat-datepicker-toggle>

    <mat-datepicker #picker></mat-datepicker>

    <mat-error \*ngIf="f.releaseDate.touched && f.releaseDate.errors">Release date required</mat-error>

  </mat-form-field>

  <mat-form-field class="f-w">

    <mat-label>Brand</mat-label>

    <mat-select formControlName="brandId">

      <mat-option \*ngFor="let b of brands" [value]="b.brandId">

**{{**b.brandName**}}**

      </mat-option>

    </mat-select>

    <mat-error \*ngIf="f.brandId.touched && f.brandId.errors">Brand required</mat-error>

  </mat-form-field>

  <div class="pic-row">

    <div class="f-img"><img \*ngIf="product != null" src="/Uploads/**{{**product.picture**}}**" width="40" /></div>

    <mat-form-field appearance="standard" class="f-w-1">

      <mat-label>Product picture</mat-label>

      <ngx-mat-file-input formControlName="picture" [accept]="'image/\*'" (change)="onChange($event)"></ngx-mat-file-input>

      <mat-icon matSuffix>insert\_photo</mat-icon>

      <mat-error \*ngIf="f.picture.touched && f.picture.errors">Picture required</mat-error>

    </mat-form-field>

  </div>

  <div class="f-w">

    <mat-label>Discontinued</mat-label>

    <mat-checkbox formControlName="discontinued" style="margin-left:15px;">

    </mat-checkbox>

    <mat-error \*ngIf="f.discontinued.touched && f.discontinued.errors">Price required</mat-error>

  </div>

  <div class="f-w">

    <button [disabled]="productForm.invalid" mat-raised-button type="submit" color="accent"><mat-icon>save\_alt</mat-icon> Save</button>

  </div>

  <div class="f-w">

    <a mat-flat-button [routerLink]="['/products']">Back to list</a>

  </div>

</form>

1. in edit-product.components.css

.form-container {

  width: 65%;

  margin: 5px auto 10px auto;

}

.f-w {

  width: 100%;

  margin-bottom: .5rem;

}

.pic-row {

    display: flex;

    flex-direction:row;

    align-items:center;

    justify-content:space-between;

}

.f-w-1 {

  width: calc(100% - 50px);

  margin-bottom: .5rem;

}

.f-img {

  max-height: 40px;

  max-width: 40px;

  overflow: hidden;

  border: solid 1px #ccc;

  -webkit-border-radius: 50%;

  border-radius: 50%;

  -webkit-transition: -webkit-box-shadow 0.3s ease;

  transition: box-shadow 0.3s ease;

}

try yourself if have any problem see M9 class-11 and another project in class-12 and its tutorial inthats folder in D drive