



CJC

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

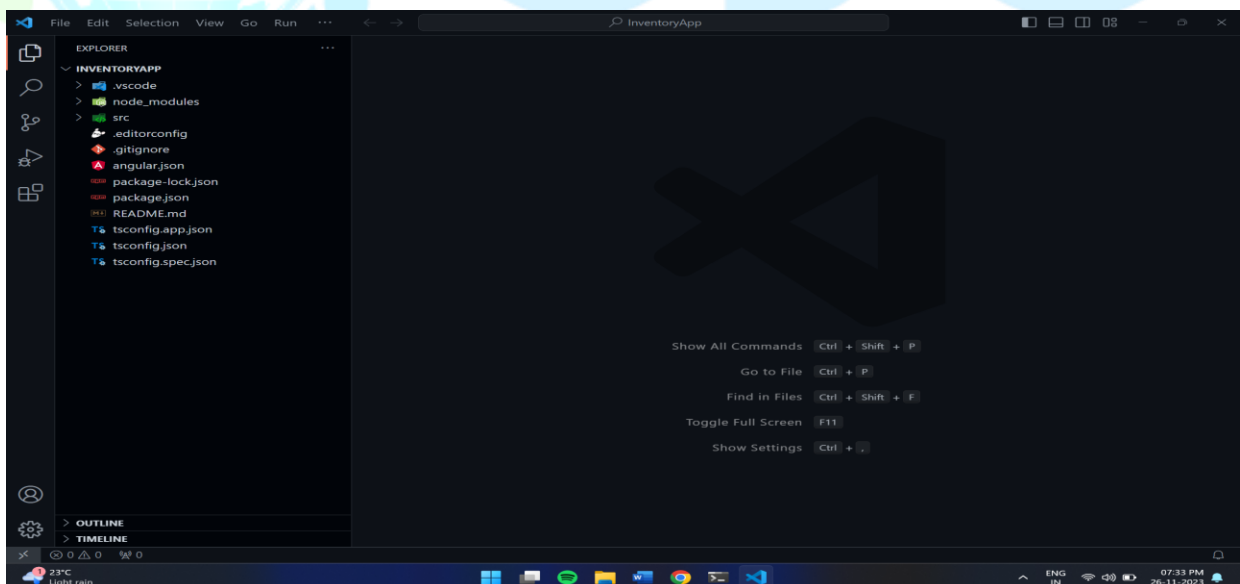
❖ Example of CRUD using routing :-

1. Create Angular project with routing as bellow.

```
npm
Microsoft Windows [Version 10.0.22621.2715]
(c) Microsoft Corporation. All rights reserved.

E:\AngularWork>ng new InventoryApp
? Would you like to add Angular routing? Yes
? Which stylesheet format would you like to use? (Use arrow keys)
> CSS
SCSS [ https://sass-lang.com/documentation/syntax#scss ]
Sass [ https://sass-lang.com/documentation/syntax#the-indented-syntax ]
Less [ http://lesscss.org ]
```

2. After project got build, open the project in visual studio code.





CJC

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

3. Install bootstrap and fontawesome library using npm commands as below.

A. For Bootstrap :- npm i bootstrap

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS E:\AngularWork\InventoryApp> npm i bootstrap
added 2 packages, and audited 1013 packages in 3s
119 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
PS E:\AngularWork\InventoryApp>
```

B. For Fontawesome :- npm i font-awesome

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS E:\AngularWork\InventoryApp> npm i font-awesome
added 1 package, and audited 1014 packages in 2s
119 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
PS E:\AngularWork\InventoryApp>
```



by Kunal Sir

Angular CRUD With Routing

- Now add the path of bootstrap.css and font-awesome.css files from their respective libraries in angular.json file as below.

The screenshot shows the VS Code editor with the `angular.json` file open. The `styles` array is highlighted with a green box, showing the addition of `src/styles.css`, `node_modules/bootstrap/dist/css/bootstrap.css`, and `node_modules/font-awesome/css/font-awesome.css`. The terminal window shows the command `npm i font-awesome` being executed, resulting in the installation of the package.

```
angular.json
19 "polyfills": [
20   "zone.js"
21 ],
22 "tsConfig": "tsconfig.app.json",
23 "assets": [
24   "src/favicon.ico",
25   "src/assets"
26 ],
27 "styles": [
28   "src/styles.css",
29   "node_modules/bootstrap/dist/css/bootstrap.css",
30   "node_modules/font-awesome/css/font-awesome.css"
31 ],
32 "scripts": [
33 ],
34 "configurations": {
35   "production": {
36     "budgets": [
```

```
PS E:\AngularWork\InventoryApp> npm i font-awesome
added 1 package, and audited 1014 packages in 2s

119 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS E:\AngularWork\InventoryApp>
```

- Now, install json-server using npm command : `npm i json-server -g`

The screenshot shows the VS Code terminal with the command `npm i json-server -g` being executed. The output shows that 116 packages were changed and 15 packages are looking for funding.

```
PS E:\AngularWork\InventoryApp> npm i json-server -g
changed 116 packages in 8s

15 packages are looking for funding
  run `npm fund` for details
PS E:\AngularWork\InventoryApp>
```



by Kunal Sir

Angular CRUD With Routing

6. Create Json document as 'product.json' in src/assets directory and add one dummy record as below.

```
{ } product.json ×
src > assets > { } product.json > [ ] product
1 {
2   "product":[
3     {
4       "id":1,
5       "productName":"break pad",
6       "manufacturer":"TATA",
7       "availableQuantity":500,
8       "reorderLevel":"High",
9       "color":"Default",
10      "details":"hydraulic disk break ",
11      "supplier":{
12        "supplierName":"Mr. Vaid",
13        "supplierCompanyName":"Vaid Automobile parts limited",
14        "supplierEmail":"vaidk@gmail.com",
15        "contactNumber":9876543210,
16        "Address":"Near warje petrol pump",
17        "city":"pune",
18        "pinCode":"411058",
19        "state":"Maharashtra"
20      }
21    }
22  ]
23 }
24 }
```

7. Now create data model (entity) according to the product.json document.

Note:- In above product.json file we can see product have the reference of supplier, so we need to create two model classes, one for product and one for supplier .



CJC

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

- A. Create Supplier class using command: `ng g class model/supplier --skip-tests`, & and data members as below.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS E:\AngularWork\InventoryApp> ng g class model/supplier --skip-tests
```

```
TS supplier.ts X
src > app > model > TS supplier.ts > Supplier
1  export class Supplier {
2
3      supplierName:string;
4      supplierCompanyName:string;
5      supplierEmail:string;
6      contactNumber:number;
7      Address:string;
8      city:string;
9      pinCode:number;
10     state:string;
11
12
13 }
14
```



CJC

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

B. Create Product class using command: `ng g class model/product --skip-tests`

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + - [ ] [ ] ... ^ x
● PS E:\AngularWork\InventoryApp> ng g class model/product --skip-tests
  CREATE src/app/model/product.ts (25 bytes)
○ PS E:\AngularWork\InventoryApp>
```

Here we need to add the reference of supplier class as shown below.

```
TS product.ts x
src > app > model > TS product.ts > ...
1  import { Supplier } from "./supplier";
2
3  export class Product {
4
5      id:number;
6      productName:string;
7      manufacturer:string;
8      availableQuantity:number;
9      reoderLevel:string;
10     color:string;
11     details:string;
12     supplier:Supplier;
13 }
14
```

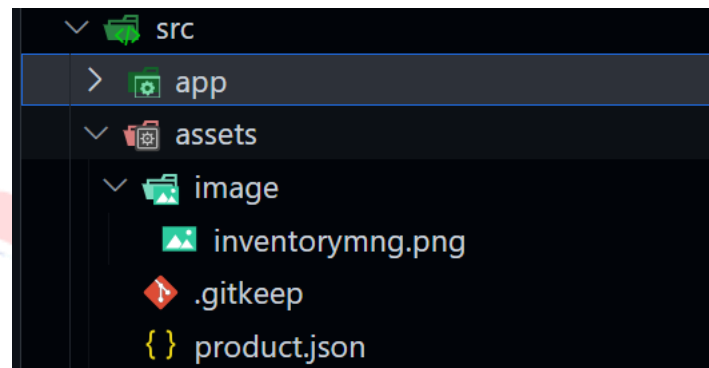


by Kunal Sir

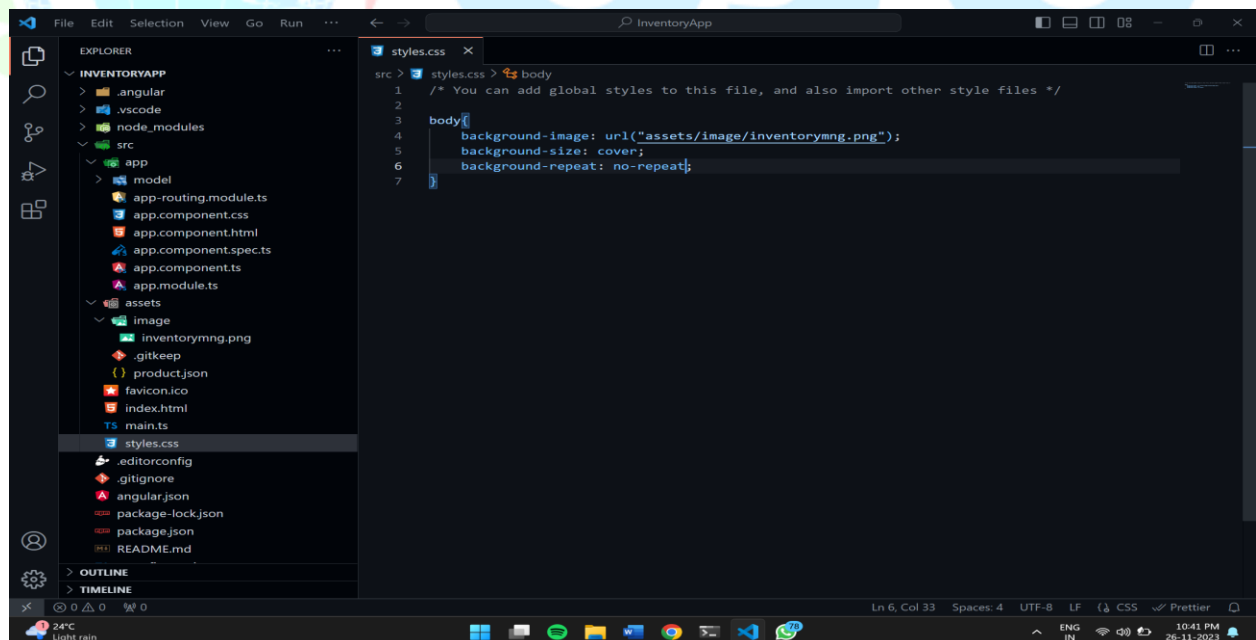
Angular CRUD With Routing

8. To apply background image, download the image from below link (you can use different image to)
- <https://adynamics.com.my/wp-content/uploads/2022/06/what-is-inventory-management-techniques-practices-benefits-.png>

9. Paste the download image in src/assets/image directory from file explorer.



10. Now in styles.css file set this image as a background-image to body element.





CJC

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

11. Create Header component using command: `ng g c template/header --skip-tests`

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + - [ ] [ ] ... ^ x
● PS E:\AngularWork\InventoryApp> ng g c template/header --skip-tests
  CREATE src/app/template/header/header.component.html (21 bytes)
  CREATE src/app/template/header/header.component.ts (202 bytes)
  CREATE src/app/template/header/header.component.css (0 bytes)
  UPDATE src/app/app.module.ts (484 bytes)
○ PS E:\AngularWork\InventoryApp>
```

12. Now add header component's selector in app.component.html file.

```
app.component.html x
src > app > app.component.html > app-header
1 <app-header></app-header>
```



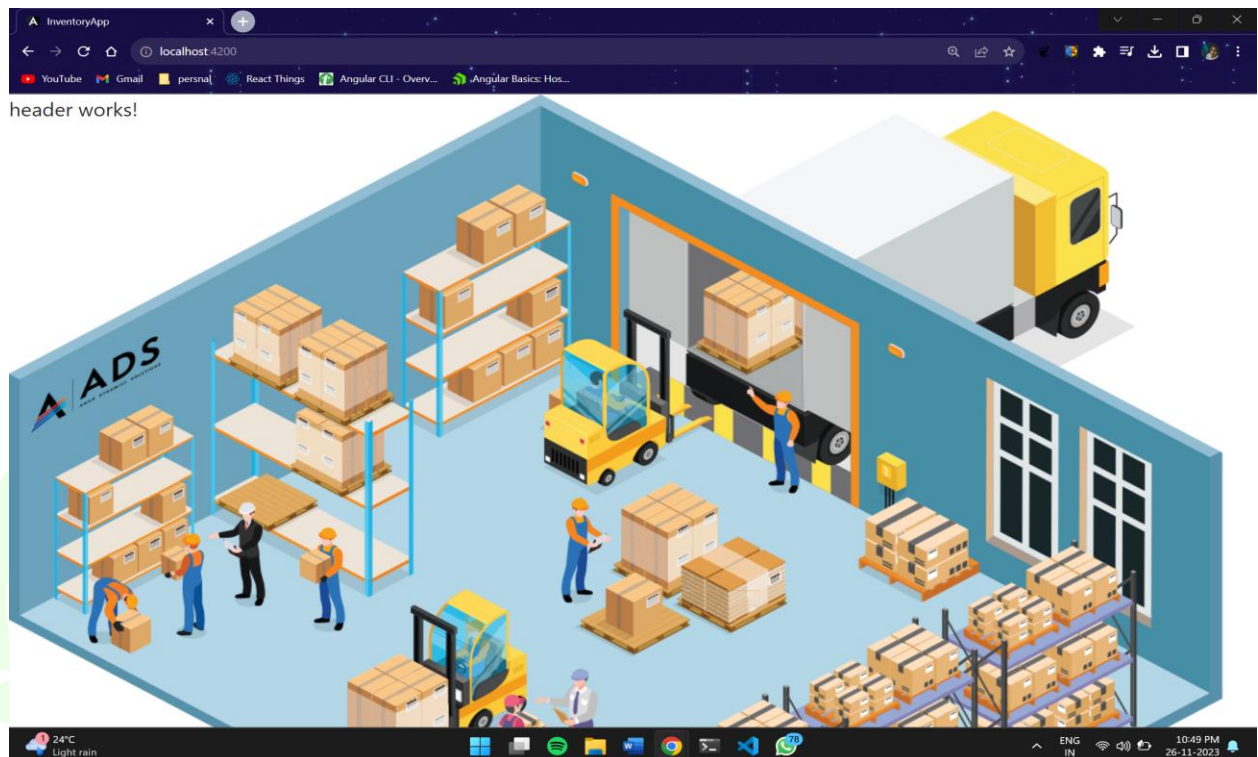

CJC

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

13. Check out-put on browser.



14. Now in header.component.html create header as below.

```
header.component.html X
src > app > template > header > header.component.html > div.heading
1  <div class="heading">
2
3    
4
5    <h1 class="slogan">
6      Taking your bike performance<br> to the next level...!
7    </h1>
8
9  </div>
```



CJC

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

15. In header.component.css write css for header and slogan as bellow.

```
header.component.css X
src > app > template > header > header.component.css > .slogan
1  .heading{
2      background-color: yellow;
3      display: flex;
4      justify-content: space-between;
5      height: 70px;
6  }
7  .slogan{
8      color: red;
9      font-size: x-large;
10     font-family: cursive;
11     font-weight: bold;
12     margin-right: 100px;
13     -webkit-text-stroke-width: 1px;
14     -webkit-text-stroke-color: black;
15 }
```

16. Create inventory module with routing file using command : ng g m inventory --routing

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS E:\AngularWork\InventoryApp> ng g m inventory --routing
CREATE src/app/inventory/inventory-routing.module.ts (252 bytes)
CREATE src/app/inventory/inventory.module.ts (292 bytes)
PS E:\AngularWork\InventoryApp>
```



by Kunal Sir

Angular CRUD With Routing

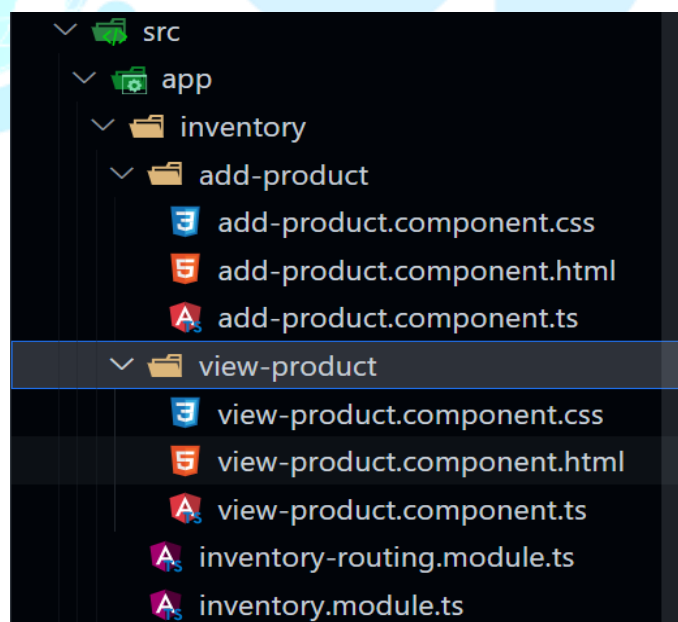
17. Create AddProductComponent and viewProductComponent in inventory module as below.

a. For AddProductComponent : `ng g c inventory/add-product --skip-tests`

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS E:\AngularWork\InventoryApp> ng g c inventory/add-product --skip-tests
CREATE src/app/inventory/add-product/add-product.component.html (26 bytes)
CREATE src/app/inventory/add-product/add-product.component.ts (221 bytes)
CREATE src/app/inventory/add-product/add-product.component.css (0 bytes)
UPDATE src/app/inventory/inventory.module.ts (498 bytes)
PS E:\AngularWork\InventoryApp>
```

b. For viewProductComponent: `ng g c inventory/view-product --skip-tests`

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS E:\AngularWork\InventoryApp> ng g c inventory/view-product --skip-tests
CREATE src/app/inventory/view-product/view-product.component.html (27 bytes)
CREATE src/app/inventory/view-product/view-product.component.ts (225 bytes)
CREATE src/app/inventory/view-product/view-product.component.css (0 bytes)
UPDATE src/app/inventory/inventory.module.ts (398 bytes)
PS E:\AngularWork\InventoryApp>
```





by Kunal Sir

Angular CRUD With Routing

18. In app.component.html add two buttons as below. Also write router-outlet element as below.

```
app.component.html X
src > app > app.component.html > router-outlet
1 <app-header></app-header>
2
3 <div class="text-center">
4   <button class="btn btn-warning me-5" routerLink="inventory/add">Add Product</button>
5   <button class="btn btn-warning" routerLink="inventory/view">View Product</button>
6 </div>
7 <router-outlet></router-outlet>
8
```

❖ Lazy Loading :-

While developing single page application we need create multiple component some of them are design to perform corresponding or inter-relational activities such components we have to declare in custom modules.

In real-time applications each module is heavy ,which means they contains large number of components , where each component have huge line of code.

In Angular, NgModules are eagerly loaded by default, it means as soon as your application renders on browser window, all the custom modules and components which are declared inside those modules are directly loaded on browser whether they are initially required or not. Due to this the performance of our application get decreed.

To avoid such kind of performance issues we use lazy loading concept in our application. Lazy loading helps to keep initial bundle size smaller. which reduces initial loading time of a application, and then after the only that NgModule will get load on browser, which end user wants to render.

19. In app-routing.module.ts file, lazily load the inventory module on 'inventory' path.



CJC

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

```
app-routing.module.ts X
src > app > app-routing.module.ts > AppRoutingModuleModule
1  import { NgModule } from '@angular/core';
2  import { RouterModule, Routes } from '@angular/router';
3
4  const routes: Routes = [
5    {path:'inventory' ,
6      loadChildren:()=>import('src/app/inventory/inventory.module').then(m=>m.InventoryModule)
7    }
8  ];
9
10 @NgModule({
11   imports: [RouterModule.forRoot(routes)],
12   exports: [RouterModule]
13 })
14 export class AppRoutingModuleModule { }
15
```

20. In inventory-routing.module.ts file map the 'add' and 'view' endpoints of the url.

```
inventory-routing.module.ts X
src > app > inventory > inventory-routing.module.ts > routes > component
1  import { NgModule } from '@angular/core';
2  import { RouterModule, Routes } from '@angular/router';
3  import { AddProductComponent } from './add-product/add-product.component';
4  import { ViewProductComponent } from './view-product/view-product.component';
5
6  const routes: Routes = [
7    {
8      path:'add' , component:AddProductComponent
9    },
10   {
11     path:'view' , component:ViewProductComponent
12   },
13 ];
14
15 @NgModule({
16   imports: [RouterModule.forChild(routes)],
17   exports: [RouterModule]
18 })
19
20 export class InventoryRoutingModule { }
21
```



by Kunal Sir

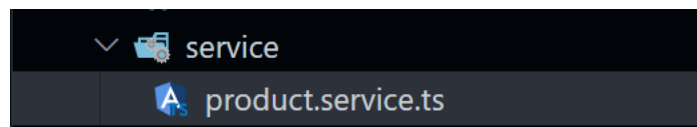
Angular CRUD With Routing

21. Check your out-put, is add-product and view-product components are rendering on browser.
22. Import ReactiveFormsModule in InventoryModule.

```
inventory.module.ts X
src > app > inventory > inventory.module.ts > InventoryModule
1 import { NgModule } from '@angular/core';
2 import { CommonModule } from '@angular/common';
3
4 import { InventoryRoutingModule } from './inventory-routing.module';
5 import { ViewProductComponent } from './view-product/view-product.component';
6 import { AddProductComponent } from './add-product/add-product.component';
7 import { ReactiveFormsModule } from '@angular/forms';
8
9
10 @NgModule({
11   declarations: [
12     ViewProductComponent,
13     AddProductComponent
14   ],
15   imports: [
16     CommonModule,
17     InventoryRoutingModule,
18     ReactiveFormsModule
19   ],
20 })
21 export class InventoryModule { }
```

23. Before implementing AddProductComponent, create and implement ProductService.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS E:\AngularWork\InventoryApp> ng g s service/product --skip-tests
CREATE src/app/service/product.service.ts (136 bytes)
PS E:\AngularWork\InventoryApp>
```





CJC

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

24. Import HttpClientModule in AppModule.

```
app.module.ts
src > app > app.module.ts > ...
1  import { NgModule } from '@angular/core';
2  import { BrowserModule } from '@angular/platform-browser';
3  import { AppRoutingModule } from './app-routing.module';
4  import { AppComponent } from './app.component';
5  import { HeaderComponent } from './template/header/header.component';
6  import { HttpClientModule } from '@angular/common/http';
7
8  @NgModule({
9    declarations: [
10     AppComponent,
11     HeaderComponent
12   ],
13   imports: [
14     BrowserModule,
15     AppRoutingModule, HttpClientModule
16   ],
17   providers: [],
18   bootstrap: [AppComponent]
19 })
20 export class AppModule { }
21
```

25. Now implement ProductService as below.

```
product.service.ts
src > app > service > product.service.ts > ProductService > updateProductDetails
1  import { HttpClient } from '@angular/common/http';
2  import { Injectable } from '@angular/core';
3  import { Product } from '../model/product';
4
5  @Injectable({
6    providedIn: 'root'
7  })
8  export class ProductService {
9
10   constructor(private http: HttpClient) { }
11
12   saveProdDetails(prod: Product)
13   {
14     return this.http.post('http://localhost:3000/product', prod);
15   }
16   getProductDetails()
17   {
18     return this.http.get('http://localhost:3000/product')
19   }
20   updateProductDetails(prod: Product)
21   {
22     return this.http.put('http://localhost:3000/product/'+prod.id , prod)
23   }
24   deleteProductDetails(id: number)
25   {
26     return this.http.delete('http://localhost:3000/product/'+id)
27   }
28 }
```



by Kunal Sir

Angular CRUD With Routing

26. Modify inventory-routing.module.ts to edit records as below.

```
inventory-routing.module.ts X
src > app > inventory > inventory-routing.module.ts > routes > component
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3 import { AddProductComponent } from '../add-product/add-product.component';
4 import { ViewProductComponent } from '../view-product/view-product.component';
5
6 const routes: Routes = [
7   {
8     path: 'add', component: AddProductComponent
9   },
10  {
11    path: 'view', component: ViewProductComponent
12  },
13  {
14    path: 'edit/:data', component: AddProductComponent
15  },
16 ];
17
18 @NgModule({
19   imports: [RouterModule.forChild(routes)],
20   exports: [RouterModule]
21 })
22 export class InventoryRoutingModule { }
23
24
25
```

27. Implement AddProductComponent as below.

I. add-product.component.ts



CJC

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

```
add-product.component.ts X
src > app > inventory > add-product > add-product.component.ts > AddProductComponent > ngOnInit
10  })
11  export class AddProductComponent implements OnInit {
12
13      constructor(private fb: FormBuilder, private prodService: ProductService,
14                  private activatedRoute: ActivatedRoute
15      ){}
16      prodForm: FormGroup;
17      ngOnInit(): void {
18
19          this.prodForm=this.fb.group(
20              {
21                  id:[],
22                  productName:[],
23                  manufacturer:[],
24                  availableQuantity:[],
25                  reorderLevel:[],
26                  color:[],
27                  details:[],
28                  supplier:this.fb.group(
29                      {
30                          supplierName:[],
31                          supplierCompanyName:[],
32                          supplierEmail:[],
33                          contactNumber:[],
34                          address:[],
35                          city:[],
36                          pinCode:[],
37                          state:[]
38                      }
39                  )
40              }
41          );
42          this.patchEditValue()
43      }
44
45      onSubmit()
46      {
47          // console.log(this.prodForm.value)
48          if(this.prodForm.controls['id'].value==0)
49          {
50              (method) ProductService.saveProdDetails(prod: Product): Observable<Object>
51              this.prodService.saveProdDetails(this.prodForm.value).subscribe();
52              alert("Product Register..!")
53          }
54          else{
55              this.prodService.updateProductDetails(this.prodForm.value).subscribe();
56              alert("Product details updated..!")
57          }
58          this.prodForm.reset();
59      }
60      patchEditValue()
61      {
62          this.activatedRoute.paramMap.subscribe(
63              param=>{
64                  let prodJson:string= param.get('data')
65                  let editDeails:Product= JSON.parse(prodJson);
66                  this.prodForm.patchValue(editDeails);
67              }
68          )
69      }
70  }
```



CJC

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

II. add-product.component.html

```
add-product.component.html X
src > app > inventory > add-product > add-product.component.html > ...

1
2 <form [formGroup]="prodForm" (ngSubmit)="onSubmit()">
3 <div class="inputContainer">
4   <div class="inputGroup bg-primary mb-3">
5     <h3>Submit Product Details...<i class="fa fa-list-alt" aria-hidden="true"></i></h3>
6   </div>
7   <div class="inputGroup">
8     <div>
9       <input type="text" formControlName="productName" placeholder="Enter Product Name">
10    </div>
11    <div>
12      <input type="text" formControlName="manufacturer" placeholder="Enter Manufacturer">
13    </div>
14  </div>
15  <div class="inputGroup">
16    <div>
17      <input type="text" formControlName="availableQuantity" placeholder="Enter Available Quantity">
18    </div>
19    <div>
20      <input type="text" formControlName="reorderLevel" placeholder="Enter Reorder Level">
21    </div>
22  </div>
23  <div class="inputGroup">
24    <div>
25      <input type="text" formControlName="color" placeholder="Enter Product Color">
26    </div>
27    <div>
28      <input type="text" formControlName="details" placeholder="Enter Product Details">
29    </div>
30  </div>
31  <div formGroupName="supplier">
32    <div class="inputGroup">
33      <div>
34        <input type="text" formControlName="supplierName" placeholder="Enter Supplier Name">
35      </div>
36      <div>
37        <input type="text" formControlName="supplierCompanyName" placeholder="Enter Supplier Company Name">
38      </div>
39    </div>
40    <div class="inputGroup">
41      <div>
42        <input type="text" formControlName="supplierEmail" placeholder="Enter Supplier Email">
43      </div>
44      <div>
45        <input type="text" formControlName="contactNumber" placeholder="Enter Contact Number">
46      </div>
47    </div>
48    <div class="inputGroup">
49      <div>
50        <input type="text" formControlName="address" placeholder="Enter Supplier Address">
51      </div>
52      <div>
53        <input type="text" formControlName="city" placeholder="Enter Supplier city">
54      </div>
55    </div>
56    <div class="inputGroup">
57      <div>
58        <input type="text" formControlName="pinCode" placeholder="Enter Supplier Pincode">
59      </div>
60      <div>
61        <input type="text" formControlName="state" placeholder="Enter Supplier State">
62      </div>
63    </div>
64    <div class="inputGroup mt-3">
65      <button class="btn btn-primary hover ">Submit</button>
66    </div>
67  </div>
68 </div>
69 </div>
70 </form>
```



by Kunal Sir

Angular CRUD With Routing

III. app-product.component.css

```
add-product.component.css X
src > app > inventory > add-product > add-product.component.css > .inputContainer
1  form{
2      display: flex;
3      justify-content: center;
4
5  }
6  .inputContainer{
7      height: 400px;
8      width: 500px;
9      display: flex;
10
11      flex-direction: column;
12
13
14      border-style: groove;
15      border-color: yellow;
16      border-width: 3px;
17
18
19  }
20  .inputGroup{
21      display: flex;
22      justify-content: space-around;
23  }
24  input{
25      margin-top: 10px;
26      border-style: dashed;
27      border-width: 1px;
28      border-color: black;
29      color: black;
30      font-weight: bold;
31      width: 200px;
32      text-align: center;
33  }
34  input::placeholder{
35      color: black;
36  }
```



CJC

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

28. Implement AddProductComponent as below.

I. view-product.component.ts

```
view-product.component.ts X
src > app > inventory > view-product > view-product.component.ts > ViewProductComponent > onEdit
1  import { Component, OnInit } from '@angular/core';
2  import { Router } from '@angular/router';
3  import { Product } from 'src/app/model/product';
4  import { ProductService } from 'src/app/service/product.service';
5
6  @Component({
7    selector: 'app-view-product',
8    templateUrl: './view-product.component.html',
9    styleUrls: ['./view-product.component.css']
10 })
11 export class ViewProductComponent implements OnInit {
12
13   constructor(private prodService:ProductService,private router:Router){}
14
15   products:Product[]
16   ngOnInit(): void {
17
18     this.prodService.getProductDetails().subscribe((data:Product[])=>{
19       this.products=data;
20     })
21   }
22   onEdit(prod:Product)
23   {
24     let prodJson:string=JSON.stringify(prod);
25     this.router.navigateByUrl('/inventory/edit/'+prodJson)
26   }
27   onDelete(id:number)
28   {
29     this.prodService.deleteProductDetails(id).subscribe();
30     window.location.reload();
31   }
32 }
33
```

II. view-product.component.html





CJC

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

```
view-product.component.html X
src > app > inventory > view-product > view-product.component.html > table.table.table-hover
1 <table class="table table-hover">
2   <thead class="thead-dark">
3     <tr>
4       <th>ID</th>
5       <th>Product Name</th>
6       <th>Manufacturer</th>
7       <th>Reorder Level</th>
8       <th>Color</th>
9       <th>Supplier Name</th>
10      <th>Supplier Company</th>
11      <th>Supplier Contact</th>
12      <th>Action</th>
13    </tr>
14  </thead>
15  <tbody>
16    <tr *ngFor="let p of products">
17      <td>{{p.id}}</td>
18      <td>{{p.productName}}</td>
19      <td>{{p.manufacturer}}</td>
20      <td>{{p.reorderLevel}}</td>
21      <td>{{p.color}}</td>
22      <td>{{p.supplier.supplierName}}</td>
23      <td>{{p.supplier.supplierCompanyName}}</td>
24      <td>{{p.supplier.contactNumber}}</td>
25      <td>
26
27        <button class="btn btn-primary" (click)="onEdit(p)">EDIT</button>
28        <button class="btn btn-danger" (click)="onDelete(p.id)">DELETE</button>
29
30      </td>
31    </tr>
32  </tbody>
33 </table>
```