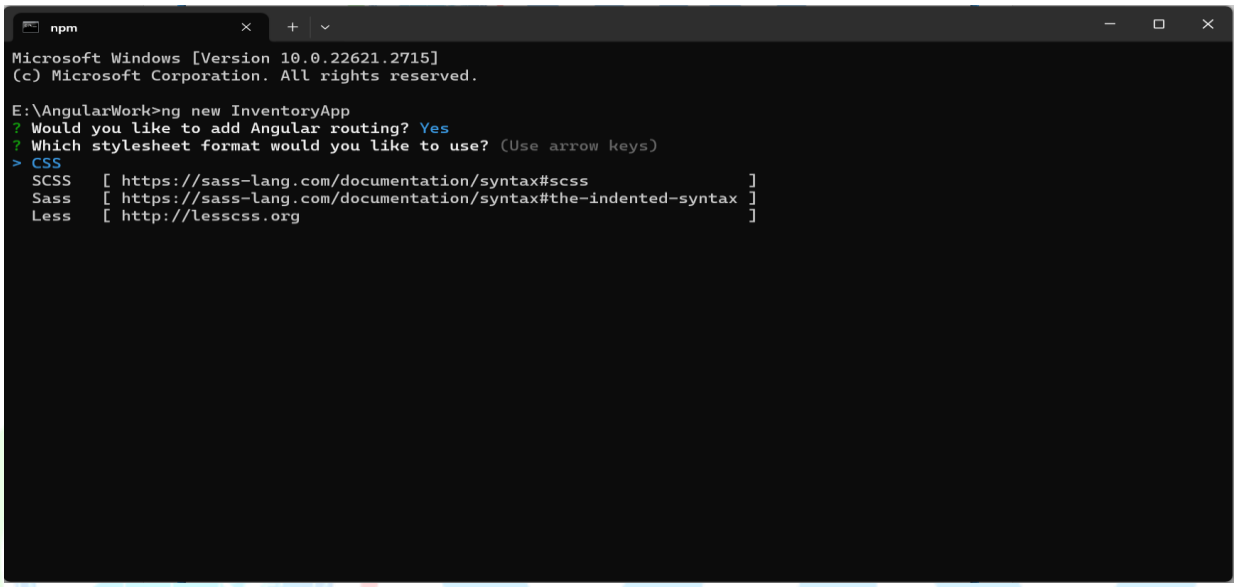
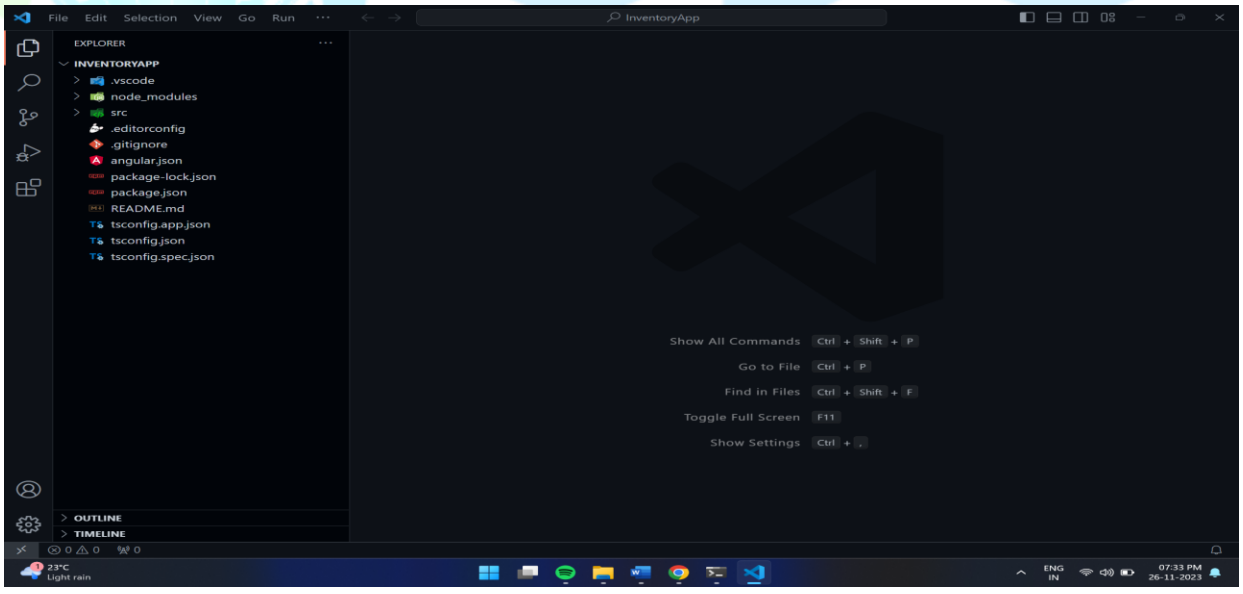
Angular CRUD With Routing

- Example of CRUD using routing :-
  1. Create Angular project with routing as bellow.



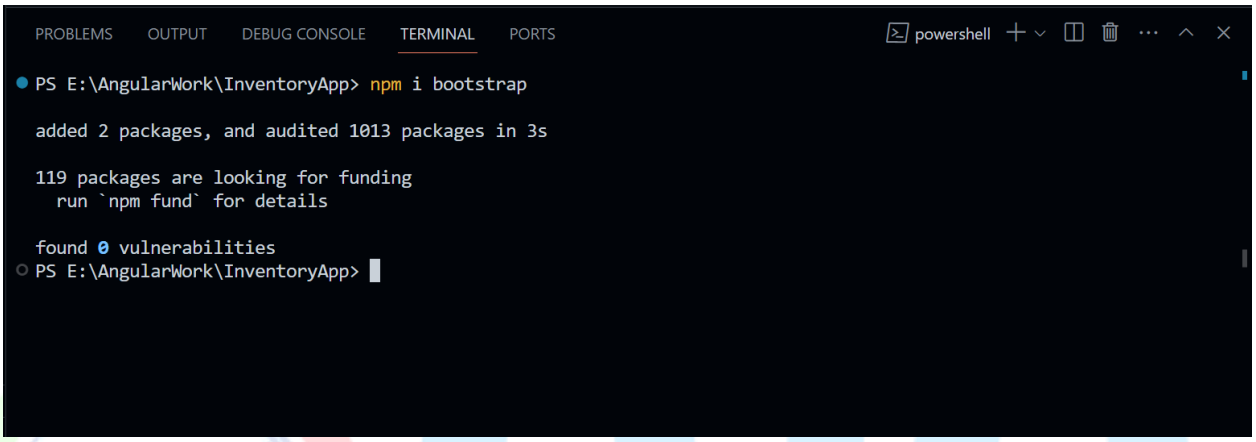
  2. After project got build, open the project in visual studio code.



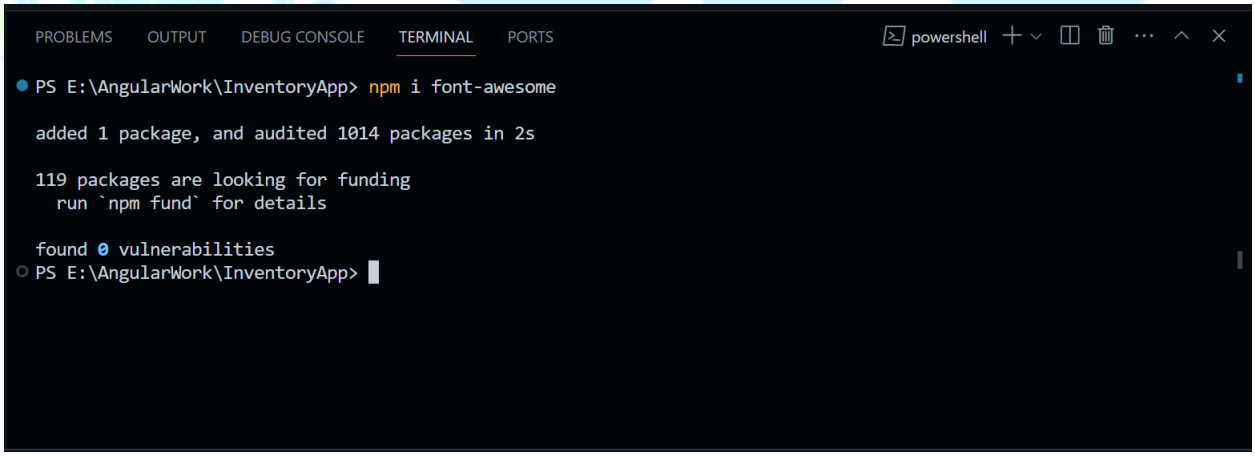
Angular CRUD With Routing

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

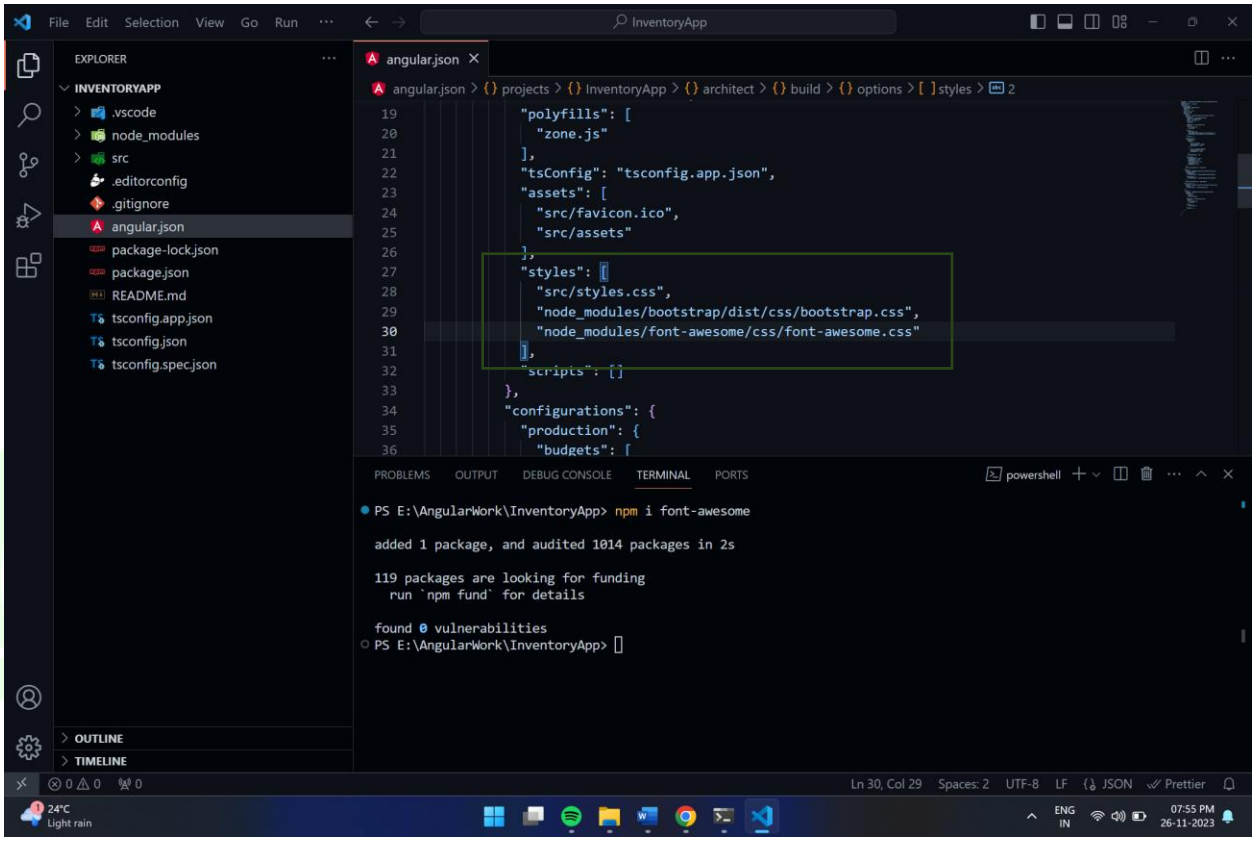
   A. For Bootstrap :- npm i bootstrap



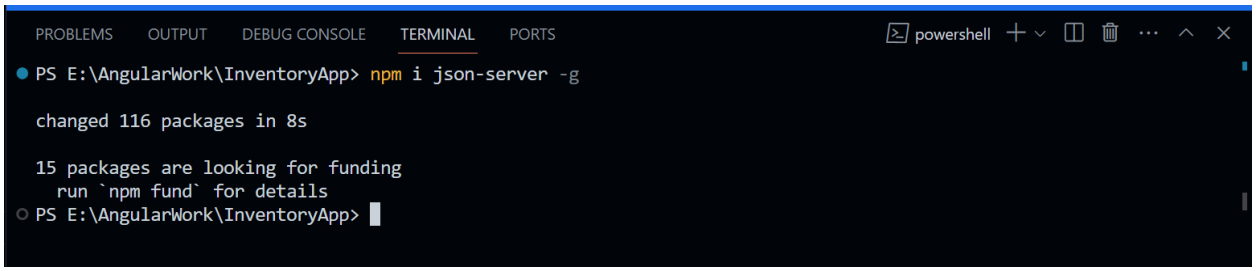
   B. For Fontawesome :- npm i font-awesome



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



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



Angular CRUD With Routing

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

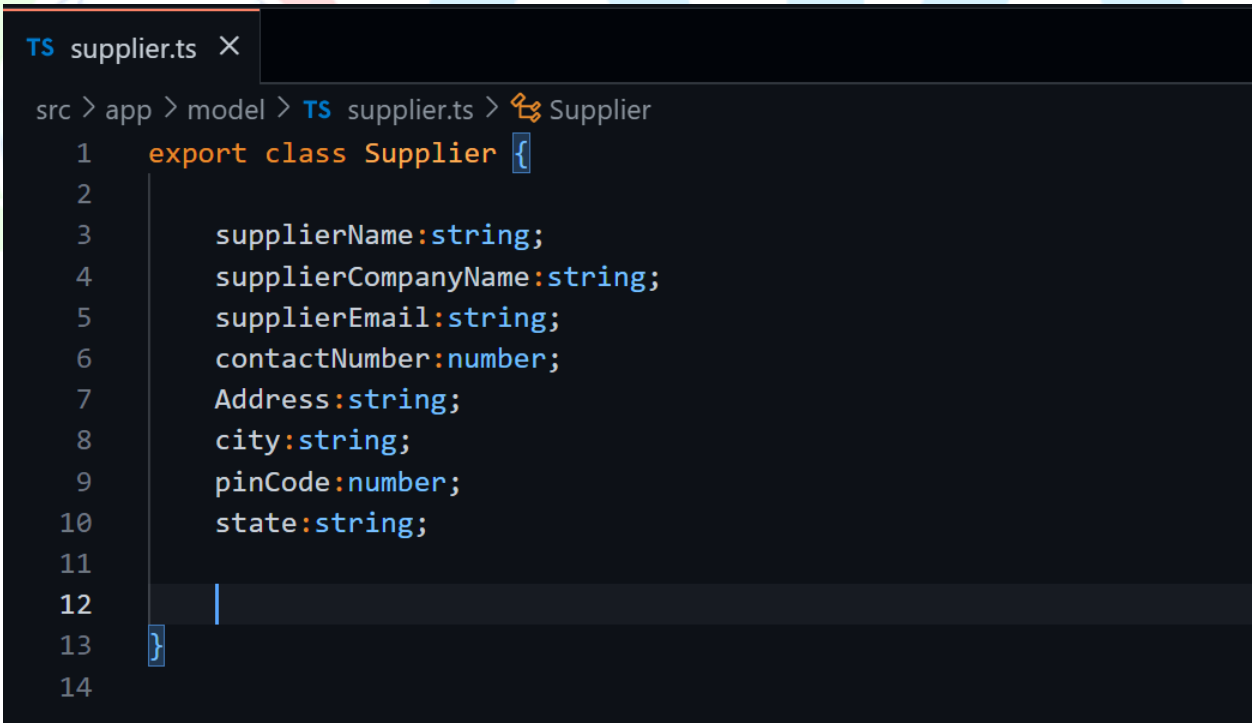
7. Now create data model (entity) according to the product.json document.

   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 .

by Kunal Sir

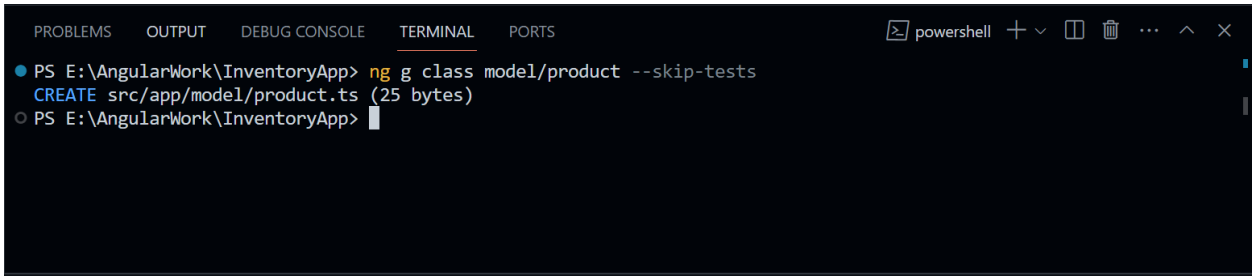
Angular CRUD With Routing

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

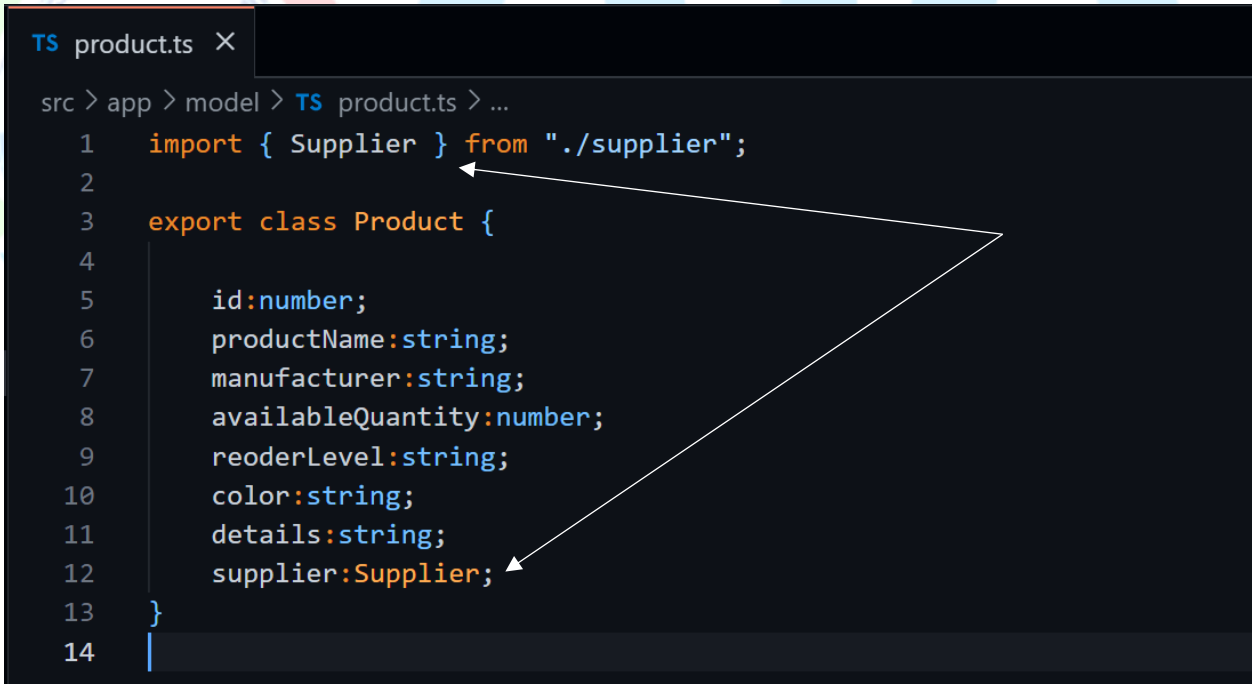


Angular CRUD With Routing

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



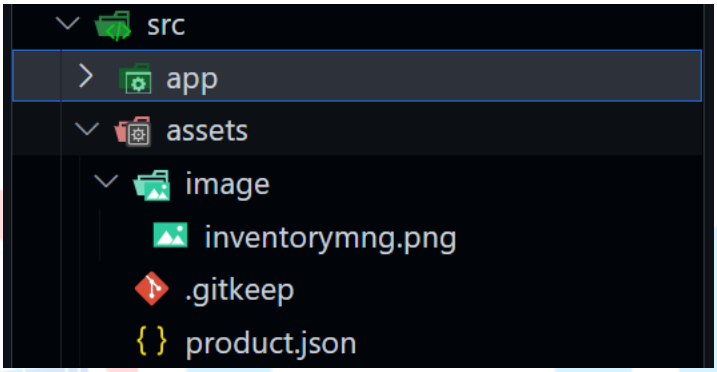
Here we need to add the reference of supplier class as shown below.



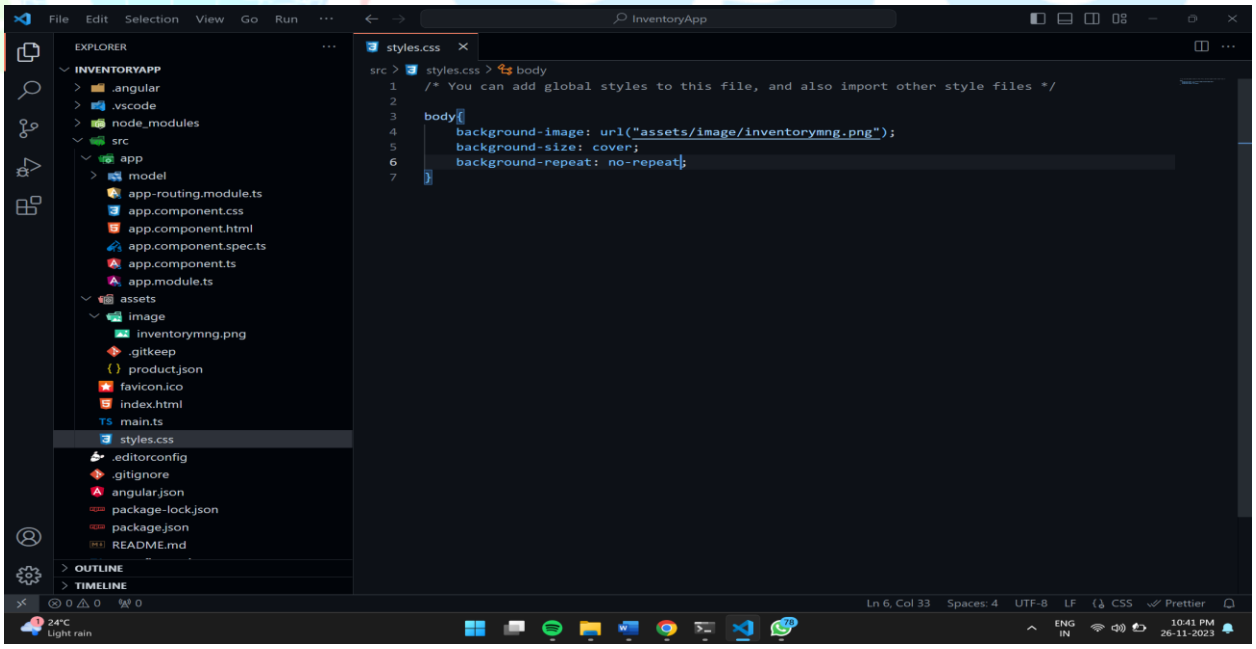
Angular CRUD With Routing

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.

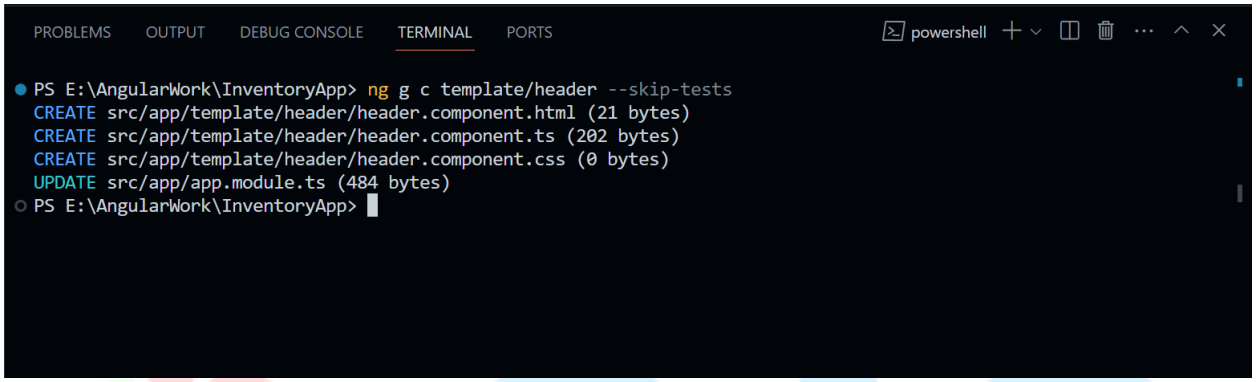


10. Now in styles.css file set this image as a background-image to body element.

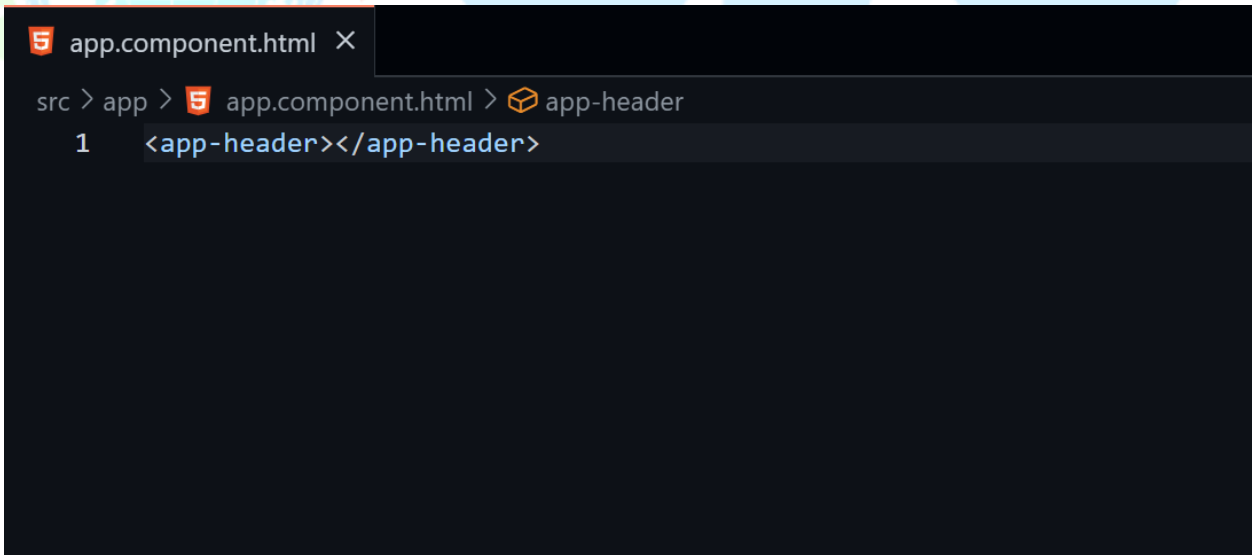


Angular CRUD With Routing

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

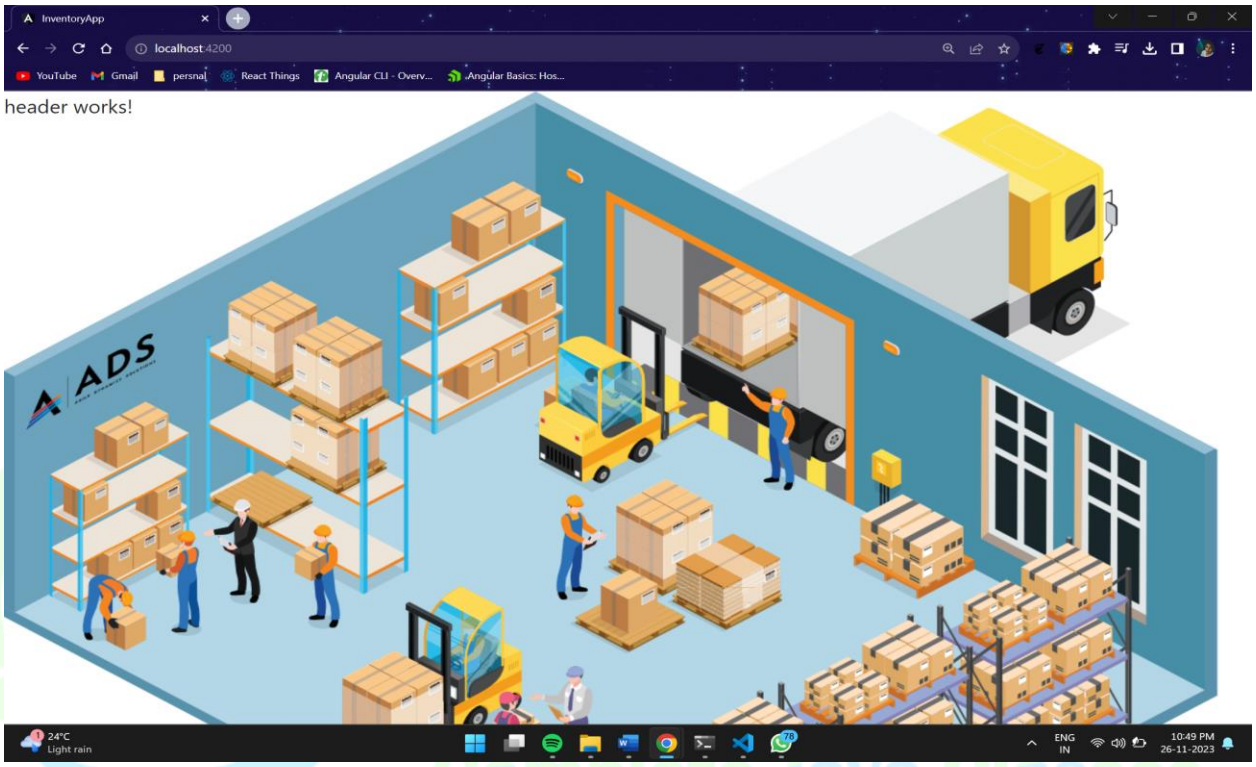


12. Now add header component's selector in app.component.html file.



Angular CRUD With Routing

13. Check out-put on browser.



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

header.component.html

src > app > template > header > header.component.html > div.heading

```
<div class="heading">

    <img src="assets/image/bikelogo.png" height="80px" width="150px">

    <h1 class="slogan">
        Taking your bike performance<br> to the next level...!
    </h1>

</div>
```

Angular CRUD With Routing

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

header.component.css

src > app > template > header > header.component.css > .slogan

```css
.heading{
    background-color: yellow;
    display: flex;
    justify-content: space-between;
    height: 70px;
}
.slogan{
    color: red;
    font-size: x-large;
    font-family: cursive;
    font-weight: bold;
    margin-right: 100px;
    -webkit-text-stroke-width: 1px;
    -webkit-text-stroke-color: black;
}
```

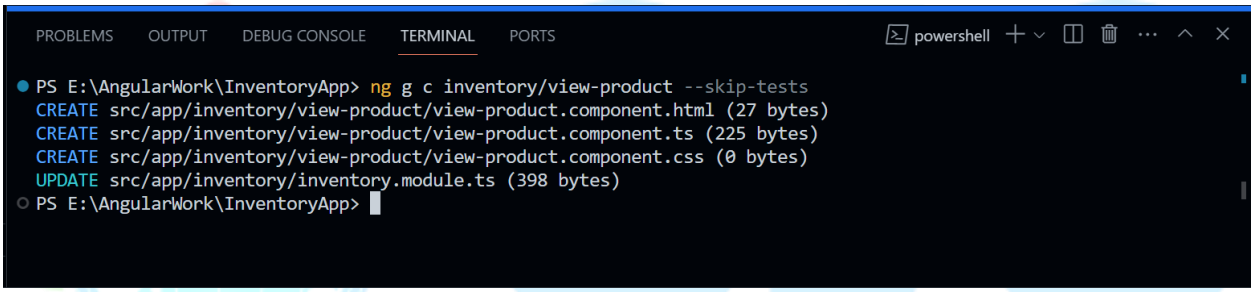
16. Create inventory module with routing file using command : ng g m inventory –routing

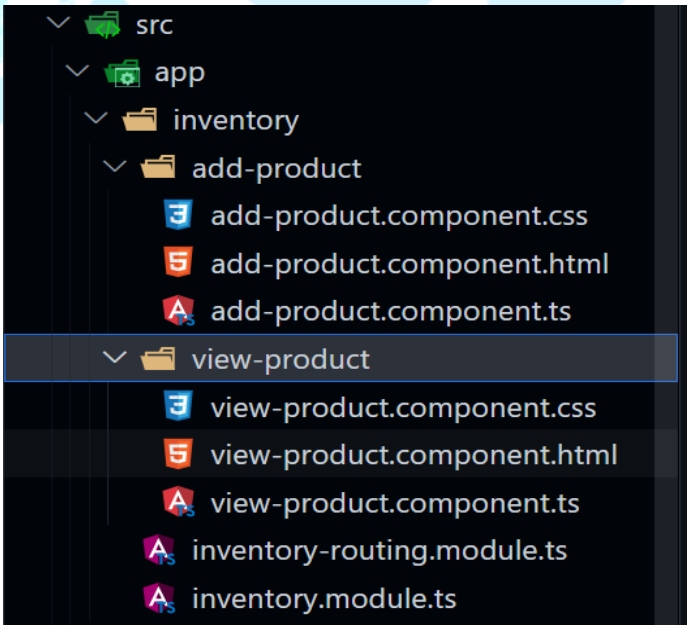
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS        powershell

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>
```

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

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





Angular CRUD With Routing

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

app.component.html

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.

Angular CRUD With Routing

app-routing.module.ts

src > app > app-routing.module.ts > AppRoutingModule

```ts
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';

const routes: Routes = [
  {path:'inventory' ,
  loadChildren:()=>import('src/app/inventory/inventory.module').then(m=>m.InventoryModule)
  }
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }

```

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

inventory-routing.module.ts

src > app > inventory > inventory-routing.module.ts > routes > component

```ts
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { AddProductComponent } from './add-product/add-product.component';
import { ViewProductComponent } from './view-product/view-product.component';

const routes: Routes = [
  {
    path:'add' , component:AddProductComponent
  },
  {
    path:'view' , component:ViewProductComponent
  }

];

@NgModule({
  imports: [RouterModule.forChild(routes)],
  exports: [RouterModule]
})
export class InventoryRoutingModule { }

```

Angular CRUD With Routing

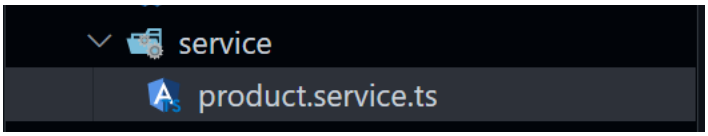
21. Check your out-put, is add-product and view-product components are rendering on browser.
22. Import ReactiveFormsModule in InventoryModule.

inventory.module.ts

```
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        powershell
PS E:\AngularWork\InventoryApp> ng g s service/product --skip-tests
CREATE src/app/service/product.service.ts (136 bytes)
PS E:\AngularWork\InventoryApp>
```



Angular CRUD With Routing

24. Import HttpClientModule in AppModule.

app.module.ts

```
src > app > app.module.ts > ...
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { HeaderComponent } from './template/header/header.component';
import { HttpClientModule } from "@angular/common/http";

@NgModule({
  declarations: [
    AppComponent,
    HeaderComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,HttpClientModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }

```

25. Now implement ProductService as below.

product.service.ts

```
src > app > service > product.service.ts > ProductService > updateProductDetails
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Product } from '../model/product';

@Injectable({
  providedIn: 'root'
})
export class ProductService {

  constructor(private http:HttpClient) { }

  saveProdDetails(prod:Product)
  {
    return this.http.post('http://localhost:3000/product',prod);
  }
  getProductDetails()
  {
    return this.http.get('http://localhost:3000/product')
  }
  updateProductDetails(prod:Product)
  {
   return this.http.put('http://localhost:3000/product/'+prod.id , prod)
  }
  deleteProductDetails(id:number)
  {
    return this.http.delete('http://localhost:3000/product/'+id)
  }
}
```

Angular CRUD With Routing

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

inventory-routing.module.ts

src > app > inventory > inventory-routing.module.ts > routes > component

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { AddProductComponent } from './add-product/add-product.component';
import { ViewProductComponent } from './view-product/view-product.component';

const routes: Routes = [
  {
    path:'add' , component:AddProductComponent
  },
  {
    path:'view' , component:ViewProductComponent
  },
  {
    path:'edit/:data' , component:AddProductComponent
  }


];

@NgModule({
  imports: [RouterModule.forChild(routes)],
  exports: [RouterModule]
})
export class InventoryRoutingModule { }

```

27. Implement AddProductComponent as below.
    I. add-product.comonent.ts

by Kunal Sir

Angular CRUD With Routing

add-product.component.ts

src > app > inventory > add-product > add-product.component.ts > AddProductComponent > ngOnInit

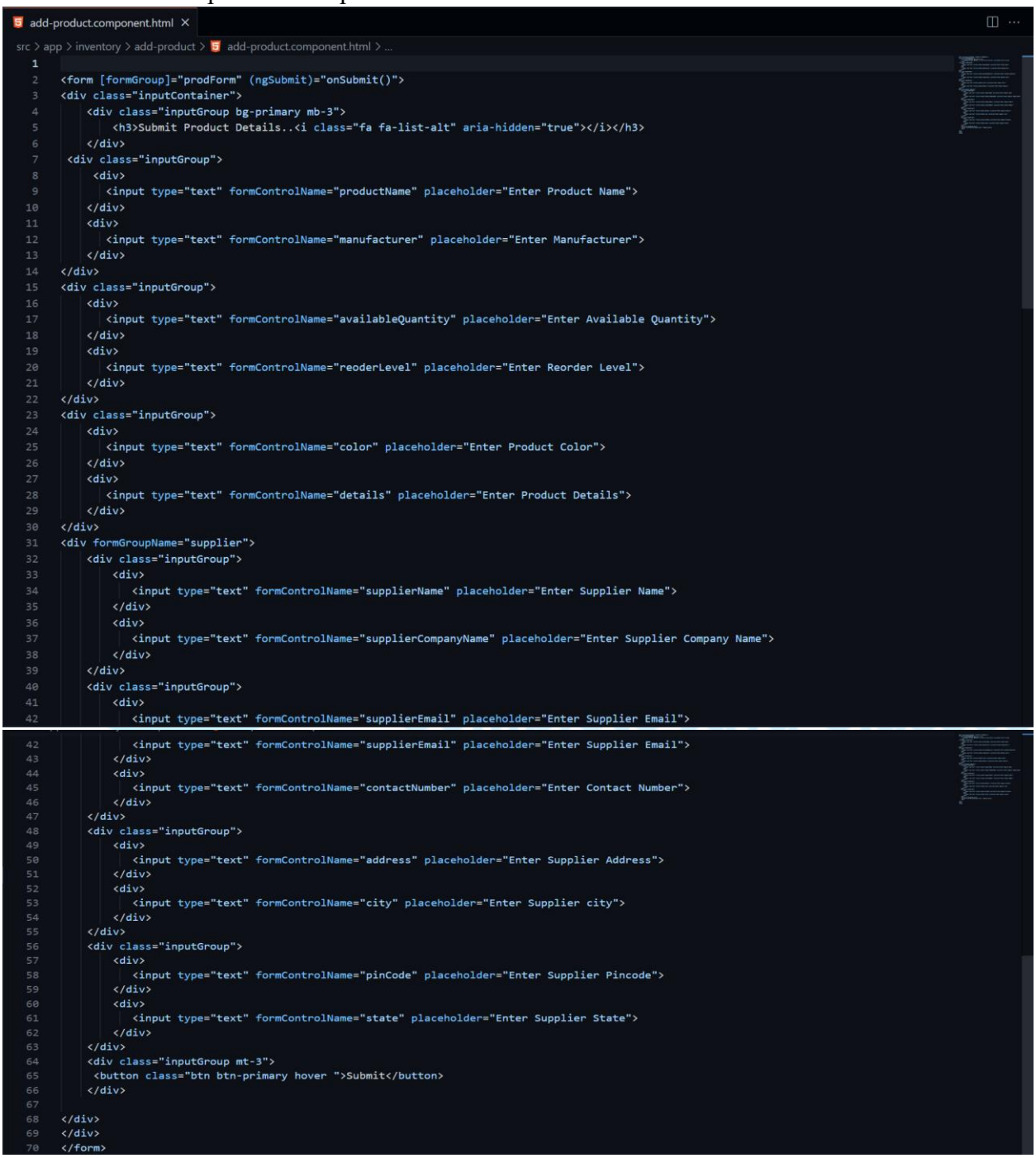
```
10  })
11  export class AddProductComponent implements OnInit {
12
13    constructor(private fb:FormBuilder, private prodService:ProductService,
14    private activatedRoute:ActivatedRoute
15      ){}
16    prodForm:FormGroup;
17    ngOnInit(): void {
18
19      this.prodForm=this.fb.group(
20        {
21          id:[],
22          productName:[],
23          manufacturer:[],
24          availableQuantity:[],
25          reoderLevel:[],
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    }
```

```
43    }
44    onSubmit()
45      {
46        // console.log(this.prodForm.value)
47          if(this.prodForm.controls['id'].value==0)
48          {                (method) ProductService.saveProdDetails(prod: Product): Observable<Object>
49        this.prodService.saveProdDetails(this.prodForm.value).subscribe();
50        alert("Product Register..!")
51          }
52          else{
53            this.prodService.updateProductDetails(this.prodForm.value).subscribe();
54            alert("Product details updated..!")
55          }
56      this.prodForm.reset();
57      }
58      patchEditValue()
59      {
60          this.activatedRoute.paramMap.subscribe(
61            param=>{
62            let prodJson:string=    param.get('data')
63            let editDeails:Product=     JSON.parse(prodJson);
64          this.prodForm.patchValue(editDeails);
65            }
66          )
67      }
68
69  }
70
```

Stop, Near, 1st Floor, Above Rupam Sweets/ Priyanka Collections Building Vikas Mitra Mandal Chowk Road, Karve Nagar, Pune, Maharashtra 411052 , Mobile No.- 8888022204

Angular CRUD With Routing

II. add-product.component.html



Angular CRUD With Routing

III. app-product.component.css

add-product.component.css

src > app > inventory > add-product > add-product.component.css > .inputContainer

```
form{
    display: flex;
    justify-content: center;

}
.inputContainer{
     height: 400px;
     width: 500px;
    display: flex;

    flex-direction: column;


  border-style: groove;
  border-color: yellow;
  border-width: 3px;


}
.inputGroup{
    display: flex;
    justify-content: space-around;
}
input{
    margin-top: 10px;
    border-style: dashed;
    border-width: 1px;
    border-color: black;
    color: black;
    font-weight: bold;
    width:200px;
    text-align: center;
}
input::placeholder{
    color: black;
}
```

Angular CRUD With Routing

28. Implement AddProductComponent as below.

I. view-product.comonent.ts

view-product.component.ts

src > app > inventory > view-product > view-product.component.ts > ViewProductComponent > onEdit

```ts
import { Component, OnInit } from '@angular/core';
import { Router } from '@angular/router';
import { Product } from 'src/app/model/product';
import { ProductService } from 'src/app/service/product.service';

@Component({
  selector: 'app-view-product',
  templateUrl: './view-product.component.html',
  styleUrls: ['./view-product.component.css']
})
export class ViewProductComponent implements OnInit{

  constructor(private prodService:ProductService,private router:Router){}

  products:Product[]
  ngOnInit(): void {

    this.prodService.getProductDetails().subscribe((data:Product[])=>{
      this.products=data;
    })
  }
  onEdit(prod:Product)
  {
        let prodJson:string=JSON.stringify(prod);
    this.router.navigateByUrl('/inventory/edit/'+prodJson)
  }
  onDelete(id:number)
  {
    this.prodService.deleteProductDetails(id).subscribe();
    window.location.reload();
  }
}

```

II. view-product.component.html

Complete Java Classes

by Kunal Sir

Angular CRUD With Routing

view-product.component.html

src > app > inventory > view-product > view-product.component.html > table.table.table-hover

```
<table class="table table-hover">
    <thead class="thead-dark">
      <tr>
       <th>ID</th>
       <th>Product Name</th>
       <th>Manufacturer</th>
       <th>Reorder Level</th>
       <th>Color</th>
       <th>Supplier Name</th>
       <th>Supplier Company</th>
       <th>Supplier Contact</th>
       <th>Action</th>
      </tr>
    </thead>
    <tbody>
        <tr *ngFor="let p of products">
          <td>{{p.id}}</td>
          <td>{{p.productName}}</td>
          <td>{{p.manufacturer}}</td>
          <td>{{p.reoderLevel}}</td>
          <td>{{p.color}}</td>
          <td>{{p.supplier.supplierName}}</td>
          <td>{{p.supplier.supplierCompanyName}}</td>
          <td>{{p.supplier.contactNumber}}</td>
          <td>

           <button class="btn btn-primary" (click)="onEdit(p)">EDIT</button>
           <button class="btn btn-danger" (click)="onDelete(p.id)">DELETE</button>

          </td>

        </tr>
    </tbody>
</table>
```