

File - C:\Users\giann\Documents\CODE\sl-project4-kitchen-story\src\main.ts

```
1 import { enableProdMode } from '@angular/core';
2 import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
3
4 import { AppModule } from '../app/app.module';
5 import { environment } from '../environments/environment';
6
7 if (environment.production) {
8   enableProdMode();
9 }
10
11 platformBrowserDynamic().bootstrapModule(AppModule)
12   .catch(err => console.error(err));
13
```

File - C:\Users\giann\Documents\CODE\sl-project4-kitchen-story\src\test.ts

```
1 // This file is required by karma.conf.js and loads recursively all the .spec and
  framework files
2
3 import 'zone.js/testing';
4 import { getTestBed } from '@angular/core/testing';
5 import {
6   BrowserDynamicTestingModule,
7   platformBrowserDynamicTesting
8 } from '@angular/platform-browser-dynamic/testing';
9
10 declare const require: {
11   context(path: string, deep?: boolean, filter?: RegExp): {
12     keys(): string[];
13     <T>(id: string): T;
14   };
15 };
16
17 // First, initialize the Angular testing environment.
18 getTestBed().initTestEnvironment(
19   BrowserDynamicTestingModule,
20   platformBrowserDynamicTesting(),
21   { teardown: { destroyAfterEach: true } },
22 );
23
24 // Then we find all the tests.
25 const context = require.context('./', true, /\.spec\.ts$/);
26 // And load the modules.
27 context.keys().map(context);
28
```

File - C:\Users\giann\Documents\CODE\sl-project4-kitchen-story\src\TO DO.txt

```
1
2 // TO DO
3
4 - Checkout
5
6 -login
```

```
1 <!doctype html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>SimplilearnKitchenStory</title>
6   <base href="/">
7   <meta name="viewport" content="width=device-width, initial-scale=1">
8   <link rel="icon" type="image/x-icon" href="favicon.ico">
9 </head>
10 <body>
11   <ks-root></ks-root>
12 </body>
13 </html>
14
```

File - C:\Users\giann\Documents\CODE\sl-project4-kitchen-story\src\styles.css

```
1  /* You can add global styles to this file, and also import other style files */
2  @import "~bootstrap/dist/css/bootstrap.min.css";
3  @import "~font-awesome/css/font-awesome.min.css";
4
```

```
1  /**
2   * This file includes polyfills needed by Angular and is loaded before the app.
3   * You can add your own extra polyfills to this file.
4   *
5   * This file is divided into 2 sections:
6   * 1. Browser polyfills. These are applied before loading ZoneJS and are sorted by
   browsers.
7   * 2. Application imports. Files imported after ZoneJS that should be loaded before
   your main
8   *     file.
9   *
10  * The current setup is for so-called "evergreen" browsers; the last versions of browsers
   that
11  * automatically update themselves. This includes Safari >= 10, Chrome >= 55 (including
   Opera),
12  * Edge >= 13 on the desktop, and iOS 10 and Chrome on mobile.
13  *
14  * Learn more in https://angular.io/guide/browser-support
15  */
16
17  /*****
   *****
18   * BROWSER POLYFILLS
19   */
20
21  /**
22   * IE11 requires the following for NgClass support on SVG elements
23   */
24  // import 'classlist.js'; // Run 'npm install --save classlist.js'.
25
26  /**
27   * Web Animations '@angular/platform-browser/animations'
28   * Only required if AnimationBuilder is used within the application and using IE/Edge or
   Safari.
29   * Standard animation support in Angular DOES NOT require any polyfills (as of Angular 6.
   0).
30   */
31  // import 'web-animations-js'; // Run 'npm install --save web-animations-js'.
32
33  /**
34   * By default, zone.js will patch all possible macroTask and DomEvents
35   * user can disable parts of macroTask/DomEvents patch by setting following flags
36   * because those flags need to be set before 'zone.js' being loaded, and webpack
37   * will put import in the top of bundle, so user need to create a separate file
38   * in this directory (for example: zone-flags.ts), and put the following flags
39   * into that file, and then add the following code before importing zone.js.
40   * import './zone-flags';
41   *
42   * The flags allowed in zone-flags.ts are listed here.
43   *
44   * The following flags will work for all browsers.
45   *
46   * (window as any).__Zone_disable_requestAnimationFrame = true; // disable patch
   requestAnimationFrame
47   * (window as any).__Zone_disable_on_property = true; // disable patch onProperty such as
   onclick
48   * (window as any).__zone_symbol__UNPATCHED_EVENTS = ['scroll', 'mousemove']; // disable
   patch specified eventNames
49   *
50   * in IE/Edge developer tools, the addEventListener will also be wrapped by zone.js
51   * with the following flag, it will bypass 'zone.js' patch for IE/Edge
52   *
53   * (window as any).__Zone_enable_cross_context_check = true;
54   */
```

```
55  */
56
57  /*****
    *****
58  * Zone JS is required by default for Angular itself.
59  */
60  import 'zone.js'; // Included with Angular CLI.
61
62
63  /*****
    *****
64  * APPLICATION IMPORTS
65  */
66
```

File - C:\Users\giann\Documents\COD\sl-project4-kitchen-story\src\app\app.module.ts

```
1 import {NgModule} from '@angular/core';
2 import {BrowserModule} from '@angular/platform-browser';
3 import {RouterModule} from '@angular/router';
4 import {AppComponent} from './app.component';
5 import {LoginComponent} from './login/login.component';
6 import {SearchComponent} from './search/search.component';
7 import {FormsModule, ReactiveFormsModule} from "@angular/forms";
8 import {HttpClientModule} from "@angular/common/http";
9 import {CartComponent} from './cart/cart.component';
10 import {CheckoutComponent} from './cart/checkout.component';
11 import {InventoryComponent} from './inventory/inventory.component';
12 import {PurchaseComponent} from './cart/purchase.component';
13 import {ChangeComponent} from './login/change.component';
14
15 @NgModule({
16   declarations: [
17     AppComponent,
18     LoginComponent,
19     SearchComponent,
20     CartComponent,
21     CheckoutComponent,
22     InventoryComponent,
23     PurchaseComponent,
24     ChangeComponent
25   ],
26   imports: [
27     BrowserModule,
28     HttpClientModule,
29     RouterModule.forRoot([
30       {path: '', component: SearchComponent},
31       {path: 'search', component: SearchComponent},
32       {path: 'login', component: LoginComponent},
33       {path: 'inventory', component: InventoryComponent},
34       {path: 'cart', component: CartComponent},
35       {path: 'checkout', component: CheckoutComponent},
36       {path: 'purchase', component: PurchaseComponent},
37       {path: 'changePassword', component: ChangeComponent},
38     ]),
39     FormsModule,
40     ReactiveFormsModule
41   ],
42   providers: [],
43   bootstrap: [AppComponent]
44 })
45 export class AppModule {
46 }
47
48
49
```



```
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'ks-root',
5   templateUrl: './app.component.html',
6   styleUrls: ['./app.component.css']
7 })
8 export class AppComponent {
9   pageTitle = 'Kitchen Story';
10
11   activeTab = 'search';
12
13   search(activeTab: string){
14     this.activeTab = activeTab;
15   }
16
17   result(activeTab: string){
18     this.activeTab = activeTab;
19   }
20 }
21
```

```
1
2 .cell {
3     border: 1px solid red;
4 }
5
6 .cell-gray {
7     border: 1px solid lightgray;
8     background-color: white;
9 }
10
11 .btn-primary, .btn-primary:hover, .btn-primary:active, .btn-primary:visited {
12
13 }
```

```

1 <div class="alert alert-info mb-1 p-1" role="alert">
2   <div class="row">
3     <div class="col-4">
4
5     </div>
6     <div class="col-4">
7       <h1 class='text-center'>
8         <span class="badge bg-info">Kitchen
9         <span class="badge bg-warning">Story</span></span>
10      </h1>
11
12    </div>
13
14    <div class="col-4 align-middle">
15      <div class="container d-flex justify-content-end">
16        <div class="p-2 align-baseline">
17          <span style="color:#A0A0A0;">Java Full Stack Bootcamp<br/>
18          Gianni Fontanot - Angular</span>
19        </div>
20      </div>
21    </div>
22  </div>
23 </div>
24
25 <div class="row" style="min-height: 100vh">
26   <div class="col-2 d-flex justify-content-end">
27     <div aria-orientation="vertical" class="nav flex-column nav-pills" id="v-
28     pills-tab" role="tablist">
29       <a [routerLink]="['/search']" class='nav-link cell-gray' routerLinkActive
30       ='active'>
31         <i aria-hidden="true" class="fa fa-search"></i>&nbsp;&nbsp;&nbsp;Search</a>
32       <a [routerLink]="['/cart']" class='nav-link cell-gray' routerLinkActive='
33       active'>
34         <i aria-hidden="true" class="fa fa-shopping-cart"></i>&nbsp;&nbsp;&nbsp;Cart
35         &nbsp;&nbsp;&nbsp;
36         <!--
37         <span class="badge bg-danger">4</span>
38         -->
39       </a>
40       <a [routerLink]="['/checkout']" class='nav-link cell-gray'
41       routerLinkActive='active'>
42         <i aria-hidden="true" class="fa fa-credit-card"></i>&nbsp;&nbsp;&nbsp;
43       Checkout</a>
44       <a [routerLink]="['/login']" class='nav-link cell-gray' routerLinkActive
45       ='active'>
46         <i aria-hidden="true" class="fa fa-sign-in"></i>&nbsp;&nbsp;&nbsp;Admin
47       login</a>
48       <a [routerLink]="['/inventory']" class='nav-link cell-gray'
49       routerLinkActive='active'>
50         <i aria-hidden="true" class="fa fa-list"></i>&nbsp;&nbsp;&nbsp;Inventory</a>
51     </div>
52   </div>
53
54   <div class="col-10">
55     <router-outlet></router-outlet>
56   </div>
57 </div>

```

File - C:\Users\giann\Documents\COD\sl-project4-kitchen-story\src\app\cart\cart.component.ts

```
1 import { Component, OnInit } from '@angular/core';
2 import { SearchService } from '../search/search.service';
3 import { FormBuilder } from '@angular/forms';
4 import { IFood } from '../model/food';
5
6 @Component({
7   templateUrl: './cart.component.html',
8   styleUrls: ['./cart.component.css']
9 })
10 export class CartComponent implements OnInit {
11   cart: IFood[] = [];
12   total: number = this.searchService.absoluteTotal;
13   constructor(private searchService: SearchService) { }
14
15   ngOnInit(): void {
16     this.cart = this.searchService.getCart();
17     this.searchService.getTotal()
18     this.total = this.searchService.getTotal()
19   }
20
21
22   removeItem(food: IFood) {
23     this.searchService.removeItem(food)
24     this.total = this.searchService.getTotal()
25   }
26 }
27
```

```

1 <div class="card">
2   <div class="card-body">
3     <div class="row cell">
4       <div class="cell col-2"></div>
5       <div class="cell col-7">
6         <div class="row cell bg-light border rounded p-1">
7           <div class="cell col-2 bg-gray"></div>
8           <div class="cell col-1">ID</div>
9           <div class="cell col-6">Food Name</div>
10          <div class="cell col-2">Price</div>
11          <div class="cell col-1"></div>
12        </div>
13        <div *ngFor="let food of cart" class="row cell border rounded p-1">
14          <div class="cell col-2 bg-gray">
15            <button (click)="removeItem(food)" class="btn btn-outline-
danger btn-sm">
16              delete
17            </button>
18          </div>
19          <div class="cell col-1">{{ food.foodId }}</div>
20          <div class="cell col-6">{{ food.foodName }}</div>
21          <div class="cell col-2">{{ food.price | currency }}</div>
22          <div class="cell col-1"></div>
23        </div>
24        <p *ngIf="cart.length<=0" class="card-text text-center mt-4"> - empty
- </p>
25      </div>
26    <div class="cell col-3"></div>
27  </div>
28 </div>
29 </div>
30

```

```

1 import {Component, OnInit} from '@angular/core';
2 import {SearchService} from "../search/search.service";
3 import {IFood} from "../model/food";
4 import {FormBuilder} from "@angular/forms";
5 import {Router} from "@angular/router";
6 import {IOrder} from "../model/order";
7
8 @Component({
9   selector: 'ks-checkout',
10  templateUrl: './checkout.component.html',
11  styleUrls: ['./checkout.component.css']
12 })
13 export class CheckoutComponent implements OnInit {
14   cart: IFood[] = []
15   cartTotal = this.searchService.absoluteTotal
16   checkoutForm = this.formBuilder.group({
17     name: '',
18     address: '',
19     credit: ''
20   });
21
22   constructor(private searchService: SearchService,
23               private formBuilder: FormBuilder,
24               private router: Router) {
25   }
26
27   ngOnInit(): void {
28     this.cart = this.searchService.getCart()
29   }
30
31   onSubmit(): void {
32     this.searchService.postPurchase(this.createOrder()).subscribe({
33       next: (data: any) => {
34         const parsed = JSON.parse(JSON.stringify(data))
35
36
37         let retVal = parsed.return
38         console.log(">>> " + retVal)
39         if (retVal === ("OK")) {
40
41           this.router.navigate(["/purchase"])
42         }
43       },
44       error: err => {
45         console.error(err.message)
46         alert("error: " + err.message);
47       }
48     });
49   }
50
51   createOrder(): IOrder {
52     this.searchService.order = {client: this.checkoutForm.value, cart: this.
searchService.cart}
53     return this.searchService.order;
54   }
55 }

```

```
1 import {Component, OnInit} from '@angular/core';
2 import {Router} from "@angular/router";
3 import {SearchService} from "../search/search.service";
4 import {IClient} from "../model/client";
5 import {IFood} from "../model/food";
6
7 @Component({
8   selector: 'ks-purchase',
9   templateUrl: './purchase.component.html',
10  styleUrls: ['./purchase.component.css']
11 })
12 export class PurchaseComponent implements OnInit {
13
14   client!: IClient;
15   cart!: IFood[];
16   absoluteTotal: number = 0;
17
18   constructor(private router: Router,
19               private searchService: SearchService) {
20   }
21
22   ngOnInit(): void {
23     this.client = this.searchService.order.client
24     this.cart = this.searchService.order.cart
25     this.absoluteTotal = this.searchService.absoluteTotal
26   }
27
28   jumpToSearch() {
29     this.router.navigate(['/search'])
30   }
31 }
32
```

File - C:\Users\giann\Documents\CODE\sl-project4-kitchen-story\src\app\cart\checkout.component.css

```
1 .cell {  
2     border: 1px solid red;  
3 }
```


File - C:\Users\giann\Documents\CODE\sl-project4-kitchen-story\src\app\cart\purchase.component.css

```
1 .cell {  
2     border: 1px solid red;  
3 }
```

```

1 <div class="row">
2   <div class="col-6">
3
4
5     <div class="card">
6       <div class="card-header">
7         Shipping
8       </div>
9       <div class="card-body">
10        <h5 class="card-title">Customer's information</h5>
11
12        <form (ngSubmit)="onSubmit()" [formGroup]="checkoutForm">
13
14          <div class="mb-2">
15            <label class="form-label mb-1" for="name">Name</label>
16            <input class="form-control" formControlName="name" id="name"
placeholder="name" type="text">
17          </div>
18          <div class="mb-2">
19            <label class="form-label mb-1" for="address">Address</label>
20            <input class="form-control" formControlName="address" id="
address" placeholder="address"
21              type="text">
22          </div>
23          <div class="mb-4">
24            <label class="form-label mb-1" for="credit">Credit Card</label>
25            <input class="form-control"
26              formControlName="credit"
27              id="credit"
28              placeholder="0000-0000-0000-0000"
29              type="text">
30          </div>
31
32          <div>
33            <button class="btn btn-primary" type="submit">Purchase</button>
34            
35          </div>
36        </form>
37      </div>
38    </div>
39    &nbsp;
40
41  </div>
42  <div class="col-6">
43
44    <div class="card">
45      <div class="card-header">
46        Total: {{cartTotal|currency}}
47      </div>
48      <div class="card-body">
49        <h5 class="card-title">Cart Content</h5>
50        <p *ngIf="cart.length<=0" class="card-text"> - empty - </p>
51
52      </div>
53
54
55      <ul *ngFor="let food of cart" class="list-group list-group-flush">
56        <li class="list-group-item">{{food.price|currency}} {{food.foodName}}</
li>
57
58      </ul>
59    </div>
60
61    <div *ngFor="let food of cart" class="row border rounded p-1
">-->

```

```
60      <!-- -->
61      <!-- <div class="row">-->
62      <!-- <div class="col-2 text-center">{{food.foodId}}</div
    >-->
63      <!-- <div class="col-8">{{food.foodName}}</div>-->
64      <!-- <div class="col-2">{{food.price|currency}}</div>-->
65      <!-- </div>-->
66      <!-- </div>-->
67      </div>
68    </div>
69  </div>
```

```

1 <div class="row">
2   <div class="col-1"></div>
3   <div class="col-9">
4     <div class="card">
5       <div class="card-header">
6         Breakdown of Your Order
7       </div>
8       <div class="card-body">
9         <h5 class="card-title mb-3">Thank you for your purchase!</h5>
10        <div>
11          <button (click)="jumpToSearch()" class="btn btn-primary mb-3">Back
to Search</button>
12        </div>
13        <div class="row">
14          <div class="col-6">
15
16            <div class="card" style="width: 18rem;">
17              <div class="card-header">
18                Client's Info
19              </div>
20              <ul class="list-group list-group-flush">
21                <li class="list-group-item">Name: {{client.name}}</li>
22                <li class="list-group-item">Address: {{client.address}}
</li>
23                <li class="list-group-item">Credit Card: {{client.credit
}}</li>
24              </ul>
25            </div>
26          </div>
27          <div class="col-6">
28
29            <div class="row bg-light border rounded p-2">
30              <div class="col-9 fw-bold">Food Name</div>
31              <div class="col-3 fw-bold">Price</div>
32            </div>
33            <div *ngFor="let food of cart" class="row border rounded p-2">
34              <div class="col-9">{{food.foodName}}</div>
35              <div class="col-3">{{food.price|currency}}</div>
36            </div>
37            <div *ngIf="cart.length>0" class="row bg-light border rounded p
-2">
38
39              <div class="col-9 fw-bold">Total</div>
40              <div class="col-3 fw-bold">{{absoluteTotal|currency}}</div>
41            </div>
42            <p *ngIf="cart.length<=0" class="card-text text-center mt-4">
- empty - </p>
43
44          </div>
45        </div>
46      </div>
47    </div>
48  </div>
49  <div class="col-2"></div>
50 </div>
51 <div class="col-2"></div>
52 </div>

```

```
1 import {Injectable} from '@angular/core';
2 import {HttpClient, HttpResponse} from "@angular/common/http";
3 import {Observable, throwError} from "rxjs";
4 import {catchError, tap} from "rxjs/operators";
5
6 @Injectable({
7   providedIn: 'root'
8 })
9 export class LoginService {
10   userUrl: string = 'https://62e8570a249bb1284ead379a.mockapi.io/api/v1/users';
11   putUrl: string = 'https://62e8570a249bb1284ead379a.mockapi.io/api/v1/users/';
12
13   constructor(private http: HttpClient) {
14   }
15
16   submitLogin(formValue: any): Observable<any> {
17     return this.http.post(this.userUrl, formValue).pipe(
18       tap(data => data),
19       catchError(err => this.handleError(err))
20     )
21   }
22 }
23
24 changePassword(formValue: any): Observable<any> {
25   return this.http.put(this.putUrl + "/" + formValue.idx, formValue).pipe(
26     tap(data => data),
27     catchError(err => this.handleError(err))
28   );
29 }
30
31 handleError(error: HttpResponse) {
32   let errorMessage
33   if (error.error instanceof ErrorEvent) {
34     errorMessage = `An error occurred: ${error.error.message}`;
35   } else {
36     errorMessage = `Server returned code: ${error.status}, error message is: ${
error.message}`;
37   }
38   console.error(errorMessage);
39   return throwError(errorMessage);
40 }
41
42 }
43
```

```
1 import {Component, OnInit} from '@angular/core';
2 import {FormBuilder} from "@angular/forms";
3 import {LoginService} from "../login.service";
4 import {Router} from "@angular/router";
5 import {InventoryService} from "../inventory/inventory.service";
6
7
8 @Component({
9   templateUrl: './login.component.html',
10   styleUrls: ['./login.component.css']
11 })
12 export class LoginComponent implements OnInit {
13
14   loginForm = this.formBuilder.group({
15     id: '1',
16     username: 'admin',
17     password: 'xadminw'
18   })
19
20
21   constructor(private formBuilder: FormBuilder,
22               private loginService: LoginService,
23               private router: Router,
24               private inventoryService: InventoryService) {
25   }
26
27   ngOnInit(): void {
28   }
29
30   onSubmit() {
31     this.loginService.submitLogin(this.loginForm.value).subscribe(
32       {
33         next: (data) => {
34           console.log(data)
35           this.inventoryService.admin = true;
36           this.router.navigate(["/inventory"])
37         },
38         error: (err => console.log(err))
39       }
40     )
41   }
42
43   onChangePassword() {
44     this.router.navigate(["/changePassword"])
45   }
46 }
47
48
```

```
1 import {Component, OnInit} from '@angular/core';
2 import {FormBuilder} from "@angular/forms";
3 import {LoginService} from "../login.service";
4 import {Router} from "@angular/router";
5
6 @Component({
7   selector: 'ks-change',
8   templateUrl: './change.component.html',
9   styleUrls: ['./change.component.css']
10 })
11 export class ChangeComponent implements OnInit {
12
13   loginGroup = this.formBuilder.group({
14     idx: '1',
15     username: 'admin',
16     oldPassword: 'admin',
17     newPassword: '',
18     newPassword2: ''
19   })
20
21
22   constructor(private formBuilder: FormBuilder,
23               private loginService: LoginService,
24               private router: Router) {
25   }
26
27   ngOnInit(): void {
28   }
29
30
31   onSubmit(): void {
32
33     if (this.loginGroup.value.newPassword !== this.loginGroup.value.newPassword2) {
34       alert("New Password does not match. Please try again.")
35     } else {
36       this.loginService.changePassword(this.loginGroup.value).subscribe({
37         next: (data: any) => {
38           const parsed = JSON.parse(JSON.stringify(data))
39           let retVal = parsed.return
40           console.log(">>> " + retVal)
41           if (retVal === ("OK")) {
42             alert("Password successfully changed")
43             this.router.navigate(["/inventory"])
44           }
45         },
46         error: err => {
47           console.error(">>> " + err)
48           alert("error: >>>" + err);
49         }
50       })
51     }
52   }
53 }
54
```

```

1 <div class="row">
2   <div class="col-2"></div>
3   <div class="col-6">
4     <div class="card">
5       <div class="card-header">
6         Welcome!
7       </div>
8       <div class="card-body">
9         <h5 class="card-title">Admin Login</h5>
10
11         <form (ngSubmit)="onSubmit()" [formGroup]="loginForm">
12           <div class="mb-3">
13             <label class="form-label" for="id">ID</label>
14             <input class="form-control" formControlName="id" id="id"
placeholder="ID"
15               type="text">
16           </div>
17           <div class="mb-3">
18             <label class="form-label" for="username">Username</label>
19             <input class="form-control" formControlName="username" id="
username" placeholder="username"
20               type="text">
21           </div>
22           <div class="mb-3">
23             <label class="form-label" for="password">Password</label>
24             <input class="form-control" formControlName="password" id="
password" placeholder="password"
25               type="password">
26           </div>&nbsp;   
27
28           <div class="row">
29             <div class="col-6 d-flex justify-content-start">
30
31               <button class="btn btn-primary" type="submit">Login</button>
32             </div>
33             <div class="col-6 d-flex justify-content-end">
34               <button (click)="onChangePassword()" class="btn btn-
secondary">Change Password</button>
35             </div>
36           </div>
37         </div>
38       </form>
39     </div>
40   </div>
41 </div>
42 <div class="col-4"></div>
43 </div>

```



```

1 <div class="row">
2   <div class="col-2"></div>
3   <div class="col-6">
4     <div class="card">
5       <div class="card-header">
6         Password Change
7       </div>
8       <div class="card-body">
9         <h5 class="card-title">Change Admin Login</h5>
10
11         <form (ngSubmit)="onSubmit()" [formGroup]="loginGroup">
12           <div class="mb-3">
13             <label class="form-label" for="idx">ID</label>
14             <input class="form-control" formControlName="idx" id="idx"
placeholder="ID"
15               type="text">
16           </div>
17           <div class="mb-3">
18             <label class="form-label" for="username">Username</label>
19             <input class="form-control" formControlName="username" id="
username"
20               placeholder="username"
21               type="text">
22           </div>
23           <div class="mb-3">
24             <label class="form-label" for="oldPassword">Old Password</
label>
25             <input class="form-control" formControlName="oldPassword" id="
oldPassword"
26               placeholder="old password"
27               type="password">
28           </div>
29           <div class="mb-3">
30             <label class="form-label" for="newPassword2">New Password</
label>
31             <input class="form-control" formControlName="newPassword" id="
newPassword"
32               placeholder="type your new password"
33               type="password">
34           </div>
35           <div class="mb-4">
36             <label class="form-label" for="newPassword2">New Password (
again)</label>
37             <input class="form-control" formControlName="newPassword2" id
="newPassword2"
38               placeholder="type again your new password"
39               type="password">
40           </div>
41
42           <button class="btn btn-primary" type="submit">Change Password</
button>
43         </form>
44       </div>
45     </div>
46   </div>
47   <div class="col-4"></div>
48 </div>
49 </div>
50

```

File - C:\Users\giann\Documents\CODE\sl-project4-kitchen-story\src\app\model\food.ts

```
1 export interface IFood {  
2   foodId: number;  
3   foodName: string;  
4   price: number;  
5 }  
6  
7  
8
```

File - C:\Users\giann\Documents\CODE\sl-project4-kitchen-story\src\app\model\user.ts

```
1 export interface IUser{  
2   name: string;  
3   password: string;  
4 }
```

File - C:\Users\giann\Documents\CODE\sl-project4-kitchen-story\src\app\model\order.ts

```
1 import {IFood} from "../food";
2 import {IClient} from "../client";
3
4 export interface IOrder {
5     client: IClient;
6     cart: IFood[];
7 }
8
```

File - C:\Users\giann\Documents\CODE\sl-project4-kitchen-story\src\app\model\client.ts

```
1 export interface IClient {  
2   name: string;  
3   address: string;  
4   credit: string;  
5 }
```

```

1 import {Injectable} from '@angular/core';
2 import {IFood} from "../model/food";
3 import {HttpClient, HttpResponse} from "@angular/common/http";
4 import {Observable, throwError} from "rxjs";
5 import {catchError, tap} from "rxjs/operators";
6 import {IOrder} from "../model/order";
7
8
9 @Injectable({
10   providedIn: 'root'
11 })
12 export class SearchService {
13   // foodUrl: string = 'api/foods.json';
14   foodUrl: string = 'https://62e8570a249bb1284ead379a.mockapi.io/api/v1/foods';
15   postPurchaseUrl: string = 'https://62e8570a249bb1284ead379a.mockapi.io/api/v1/
purchase';
16   cart: IFood[] = [];
17   order!: IOrder;
18   absoluteTotal: number = 0;
19
20   constructor(private http: HttpClient) {
21   }
22
23   getFoods(): Observable<IFood[]> {
24     return this.http.get<IFood[]>(this.foodUrl).pipe(
25       tap(data => console.log('All', JSON.stringify(data))),
26       catchError(this.handleError)
27     );
28   }
29
30   postPurchase(order: any): Observable<any> {
31     return this.http.post<any>(this.postPurchaseUrl, order).pipe(
32       tap(data => console.log(JSON.stringify(data))),
33       catchError(this.handleError)
34     );
35   }
36
37   handleError(error: HttpResponse) {
38     let errorMessage
39     if (error.error instanceof ErrorEvent) {
40       errorMessage = `An error occurred: ${error.error.message}`;
41     } else {
42       errorMessage = `Server returned code: ${error.status}, error message is: ${
error.message}`;
43     }
44     console.error(errorMessage);
45     return throwError(errorMessage);
46   }
47
48   addToCart(food: IFood) {
49     this.cart.push(food);
50   }
51
52   getCart(): IFood[] {
53     return this.cart;
54   }
55
56   removeItem(food: IFood) {
57
58     let foodIdToDelete = (element: { foodId: number; }) => food.foodId === element.
foodId;
59
60     let index = this.cart.findIndex(foodIdToDelete)
61     this.cart.splice(index, 1)

```

```
62     }
63
64     getTotal(): number {
65         this.absoluteTotal = 0;
66         this.cart.forEach(item => this.absoluteTotal += item.price)
67         return this.absoluteTotal;
68     }
69 }
70
71
```

```
1 import {Component, OnDestroy, OnInit} from '@angular/core';
2 import {IFood} from "../model/food";
3 import {SearchService} from "../search.service";
4 import {Subscription} from "rxjs";
5
6 @Component({
7   templateUrl: './search.component.html',
8   styleUrls: ['./search.component.css']
9 })
10 export class SearchComponent implements OnInit, OnDestroy {
11
12   foodsFiltered: IFood[] = [];
13   foods: IFood[] = [];
14   cart: IFood [] = []
15   listFilter: string = '';
16   sub!: Subscription;
17
18   constructor(private searchService: SearchService) {
19   }
20
21   ngOnInit(): void {
22     this.sub = this.searchService.getFoods().subscribe({
23       next: data => {
24         this.foods = data;
25         this.foodsFiltered = this.foods;
26         return this.foodsFiltered;
27       },
28       error: err => alert(err)
29     });
30   }
31
32   search() {
33     this.sub = this.searchService.getFoods().subscribe({
34       next: data => {
35         this.foods = data
36         this.foodsFiltered = this.foods.filter(
37           food => food.foodName.toLowerCase().includes(
38             this.listFilter.toLowerCase()
39           )
40         )
41       },
42       error: err => alert(err),
43     });
44   }
45
46   ngOnDestroy(): void {
47     //this.sub.unsubscribe();
48     //food => food.foodName.toLowerCase()
49     // .includes(this.listFilter.toLowerCase()
50   }
51
52   select(food: IFood) {
53     this.searchService.addToCart(food);
54   }
55 }
56
57
58
```


File - C:\Users\giann\Documents\CODE\sl-project4-kitchen-story\src\app\search\search.component.css

```
1 body {  
2     background: #0D0;  
3 }  
4  
5 .cell {  
6     border: 1px solid red;  
7 }
```

```

1 <div class="card">
2   <div class="card-header">
3     Our Products
4   </div>
5   <div class="card-body">
6     <h5 class="card-title">This is the list of foods we offer.</h5>
7     <p class="card-text">Select the foods you would like to order. When you are done
, click the Cart menu to see
8       a list of the items. You can come back to this page to add more items to your
    cart if
9       you want.</p>
10  </div>
11  <div class="row pb-3 pt-3">
12    <div class="col-3"></div>
13    <div class="col-5">
14      <input [(ngModel)]="listFilter" class="form-control" placeholder="What are
you looking for?" type="text">
15    </div>
16    <div class="col-3 ps-0 ms-0">
17      <button (click)="search()" class="btn btn-primary">search</button>
18    </div>
19  </div>
20 </div>
21 <div class="container">
22   <div class="row">
23     <div class="col-2"></div>
24     <div class="col-7">
25       <div class="row bg-light border rounded p-1">
26         <div class="col-2 bg-gray"></div>
27         <div class="col-2 text-center">ID</div>
28         <div class="col-5">Food Name</div>
29         <div class="col-2">Price</div>
30         <div class="col-1"></div>
31       </div>
32       <div *ngFor="let food of foodsFiltered" class="row border rounded p-1">
33         <div class="col-2 bg-gray">
34           <button (click)="select(food)" class="btn btn-outline-primary
btn-sm">select</button>
35         </div>
36         <div class="col-2 text-center">{{food.foodId}}</div>
37         <div class="col-5">{{food.foodName}}</div>
38         <div class="col-2">{{food.price|currency}}</div>
39         <div class="col-1"></div>
40       </div>
41     </div>
42   <div class="col-3"></div>
43 </div>
44 </div>
45 </div>

```

```

1 import {Injectable} from '@angular/core';
2 import {HttpClient, HttpResponse} from "@angular/common/http";
3 import {Observable, throwError} from "rxjs";
4 import {catchError, tap} from "rxjs/operators";
5 import {IFood} from "../model/food";
6
7 @Injectable({
8   providedIn: 'root'
9 })
10 export class InventoryService {
11   urlSave: string = "https://62e8570a249bb1284ead379a.mockapi.io/api/v1/foods";
12   urlDelete: string = "https://62e8570a249bb1284ead379a.mockapi.io/api/v1/foods";
13   urlGet: string = 'https://62e8570a249bb1284ead379a.mockapi.io/api/v1/foods';
14   admin: boolean = false;
15
16   constructor(private http: HttpClient) {
17   }
18
19   saveNewItem(inventoryGroup: any): Observable<any> {
20     return this.http.post(this.urlSave, inventoryGroup).pipe(
21       tap(data => console.log(JSON.stringify(data))),
22       catchError(this.handleError)
23     );
24   }
25
26   deleteItem(foodId: number): Observable<any> {
27     return this.http.delete(this.urlDelete + "/" + foodId).pipe(
28       tap(data => console.log("DELETE: " + JSON.stringify(data))),
29       catchError(this.handleError)
30     );
31   }
32
33   getFoods(): Observable<IFood[]> {
34     return this.http.get<IFood[]>(this.urlGet).pipe(
35       tap(data => console.log(JSON.stringify(data))),
36       catchError(this.handleError)
37     )
38   }
39
40   handleError(err: HttpResponse) {
41     let errorMessage = '';
42     if (err.error instanceof ErrorEvent) {
43       errorMessage = `An error occurred: ${err.error.message}`;
44     } else {
45       errorMessage = `Server returned code: ${err.status}, error message is: ${err.message}`;
46     }
47     console.error(errorMessage);
48     return throwError(() => errorMessage);
49   }
50 }
51

```

```

1 import {Component, OnInit} from '@angular/core';
2 import {FormBuilder} from "@angular/forms";
3 import {IFood} from "../model/food";
4 import {Subscription} from "rxjs";
5 import {InventoryService} from "../inventory.service";
6 import {Router} from "@angular/router";
7
8 @Component({
9   selector: 'ks-inventory',
10  templateUrl: './inventory.component.html',
11  styleUrls: ['./inventory.component.css']
12 })
13 export class InventoryComponent implements OnInit {
14   inventoryGroup = this.formBuilder.group({
15     foodId: '',
16     foodName: '',
17     price: ''
18   })
19
20   sub!: Subscription;
21   errMsg!: string;
22   foods: IFood[] = [];
23   result: any;
24   admin: boolean = this.inventoryService.admin;
25
26   constructor(private formBuilder: FormBuilder,
27               private inventoryService: InventoryService,
28               private router: Router) {
29   }
30
31   onSubmit(): void {
32     this.inventoryService.saveNewItem(this.inventoryGroup.value).subscribe({
33       next: (data: any) => {
34         const parsed = JSON.parse(JSON.stringify(data))
35         let retVal: string = parsed.return
36         if (retVal === ("OK")) {
37           alert("Item saved successfully.")
38           this.refreshList();
39         } else {
40           alert("error: ");
41         }
42       },
43       error: (error: any) => {
44         alert("error: " + error.message)
45       }
46     });
47   }
48
49   onDelete(item: IFood) {
50     this.inventoryService.deleteItem(item.foodId).subscribe({
51       next: (data) => {
52         const deleteOp = JSON.parse(JSON.stringify(data))
53         let retVal = deleteOp.return
54         if (retVal === ("OK")) {
55           alert("Item deleted successfully.")
56           this.refreshList();
57         } else {
58           alert("error:")
59         }
60       },
61       error: (error: any) => {
62         alert("error: " + error.message)
63       }
64     });

```

```
65     }
66
67     ngOnInit(): void {
68         this.refreshList();
69     }
70
71     refreshList(): void {
72         this.sub = this.inventoryService.getFoods().subscribe({
73             next: data => {
74                 this.foods = data
75             },
76             error: error => this.errMsg = error
77         });
78     }
79
80     ngOnDestroy(): void {
81         this.sub.unsubscribe();
82     }
83
84     jumpToLogin() {
85         this.router.navigate(["/login"])
86     }
87 }
88
```

File - C:\Users\giann\Documents\CODE\sl-project4-kitchen-story\src\applinventory\inventory.component.css

```
1 .cell {  
2     /*border: 1px solid red;*/  
3 }
```

```

1 <div *ngIf="admin==true;else other_content">
2   <div class="card-body">
3     <div class="row cell">
4       <div class="cell col-2"></div>
5       <div class="cell col-7">
6         <div class="row cell bg-light border rounded p-1">
7           <div class="cell col-2 bg-gray"></div>
8           <div class="cell col-1"></div>
9           <div class="cell col-6">Food Name</div>
10          <div class="cell col-2">Price</div>
11          <div class="cell col-1"></div>
12        </div>
13        <form (ngSubmit)="onSubmit()" [formGroup]="inventoryGroup">
14          <div class="row cell p-2">
15            <div class="cell col-2"></div>
16            <div class="cell col-1"></div>
17            <div class="cell col-6"><input class="form-control"
18              formControlName="foodName"
19              id="foodName"
20              placeholder="add a new food..."
21              size="25" type="text">
22          </div>
23          <div class="cell col-2"><input class="form-control"
24            formControlName="price"
25            id="price"
26            placeholder="0.00"
27            size="2"
28            type="text"></div>
29          <div class="cell col-1">
30            <button class="btn btn-primary" type="submit">Save</button>
31          </div>
32        </div>
33      </form>
34
35      <div *ngFor="let item of foods" class="row cell border rounded p-1">
36        <div class="cell col-2">
37          <button (click)="onDelete(item)" class="btn btn-outline-danger
38            btn-sm">Delete</button>
39          <div class="cell col-1">{{item.foodId}}</div>
40          <div class="cell col-6">{{item.foodName}}</div>
41          <div class="cell col-2">{{item.price|currency}}</div>
42          <div class="cell col-1"></div>
43        </div>
44      </div>
45    </div>
46    <div class="cell col-3"></div>
47  </div>
48 </div>
49 </div>
50 <ng-template #other_content>
51   <div class="card">
52     <h5 class="card-header">Not allowed</h5>
53     <div class="card-body">
54       <h5 class="card-title">You need to login first</h5>
55       <p class="card-text">In order to see the inventory and add or delete items,
56         first you must be looged in the
57         application.</p>
58       <a (click)="jumpToLogin()" class="btn btn-primary">Go to Login</a>
59     </div>
60   </div>
61 </ng-template>
62

```

```
1 import { ComponentFixture, TestBed } from '@angular/core/testing';
2
3 import { InventoryComponent } from './inventory.component';
4
5 describe('InventoryComponent', () => {
6   let component: InventoryComponent;
7   let fixture: ComponentFixture<InventoryComponent>;
8
9   beforeEach(async () => {
10     await TestBed.configureTestingModule({
11       declarations: [ InventoryComponent ]
12     })
13     .compileComponents();
14   });
15
16   beforeEach(() => {
17     fixture = TestBed.createComponent(InventoryComponent);
18     component = fixture.componentInstance;
19     fixture.detectChanges();
20   });
21
22   it('should create', () => {
23     expect(component).toBeTruthy();
24   });
25 });
26
```


File - C:\Users\giann\Documents\COD\sl-project4-kitchen-story\src\environments\environment.ts

```
1 // This file can be replaced during build by using the 'fileReplacements' array.
2 // 'ng build' replaces 'environment.ts' with 'environment.prod.ts'.
3 // The list of file replacements can be found in 'angular.json'.
4
5 export const environment = {
6   production: false
7 };
8
9 /*
10  * For easier debugging in development mode, you can import the following file
11  * to ignore zone related error stack frames such as 'zone.run', 'zoneDelegate.invokeTask
12  *
13  * This import should be commented out in production mode because it will have a negative
14  * impact
15  * on performance if an error is thrown.
16  */
17 // import 'zone.js/plugins/zone-error'; // Included with Angular CLI.
```

File - C:\Users\giann\Documents\CODE\sl-project4-kitchen-story\src\environments\environment.prod.ts

```
1 export const environment = {  
2   production: true  
3 };  
4
```