Services

auth.service.ts

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

@Injectable({
  providedIn: 'root',
})
export class AuthService {
  constructor() {}

  isUserLoggedIn(): Boolean {
    if (
      sessionStorage.getItem('userId') != null &&
      sessionStorage.getItem('userId') != ''
    ) {
      return true;
    }
    return false;
  }

  isAdmin(): Boolean {
    if (
      sessionStorage.getItem('userRole') != null &&
      sessionStorage.getItem('userRole') != ''
    ) {
      if (sessionStorage.getItem('userRole') == 'Admin') {
        return true;
      }
    }
    return false;
  }

  isUser(): Boolean {
    if (
      sessionStorage.getItem('userRole') != null &&
      sessionStorage.getItem('userRole') != ''
    ) {
      if (sessionStorage.getItem('userRole') == 'User') {
        return true;
      }
    }
    return false;
  }
}
```

user.service.ts

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { environment } from 'src/environments/environment';
import { UserModel } from '../model/user-model';
import { map, Observable } from 'rxjs';

@Injectable({
  providedIn: 'root',
})
export class UserService {
  private apiHost = environment.apiHost;
  private url: string;

  constructor(private httpClient: HttpClient) {}

  signingInUser(userModel: UserModel): Observable<UserModel[]> {
    this.url = `${this.apiHost}/subscribers?userId=${userModel.userId}`;
    return this.httpClient.get<UserModel[]>(this.url);
  }

  registerUser(userModel: UserModel): Observable<UserModel> {
    this.url = `${this.apiHost}/subscribers`;
    return this.httpClient.post<UserModel>(this.url, userModel);
  }

  resetPassowrd(userModel: UserModel): Observable<UserModel> {
    this.url = `${this.apiHost}/subscribers/${userModel.id}`;
    return this.httpClient.put<UserModel>(this.url,userModel);
  }
}
```

food-service.service.ts

```
import { Injectable } from '@angular/core';
import { environment } from 'src/environments/environment';
import { FoodModel } from '../model/food-model';
import { Observable } from 'rxjs';
import { HttpClient } from '@angular/common/http';
import { UserModel } from '../model/user-model';

@Injectable({
  providedIn: 'root',
})
export class FoodServiceService {
  private apiHost = environment.apiHost;
  private url: string;

  constructor(private httpClient: HttpClient) {}

  // Service Methods - AllFoodItemsComponent --------------------------------------------
---------------------
  getAllFoodItems(): Observable<FoodModel[]> {
    this.url = `${this.apiHost}/foods`;
    return this.httpClient.get<FoodModel[]>(this.url);
  }

  deleteFoodItem(id: number): Observable<FoodModel> {
    this.url = `${this.apiHost}/foods/${id}`;
    return this.httpClient.delete<FoodModel>(this.url);
  }

  // Service Methods - AddFoodItemsComponent --------------------------------------------
---------------------
  addFoodItem(foodItem: FoodModel): Observable<FoodModel> {
    this.url = `${this.apiHost}/foods`;
    return this.httpClient.post<FoodModel>(this.url, foodItem);
  }

  // Service Methods - FoodItemsComponent -----------------------------------------------
------------------
  getFoodsByCategory(category: string): Observable<FoodModel[]> {
    this.url = `${this.apiHost}/foods?foodCategory=${category}`;
    return this.httpClient.get<FoodModel[]>(this.url);
  }

  // Service Methods ------------------------------------------------------------------
  getFoodById(id: number): Observable<FoodModel> {
    let foodItem: FoodModel;
    this.url = `${this.apiHost}/foods/${id}`;
    return this.httpClient.get<FoodModel>(this.url);
  }
}
```

Models

error-enum.ts

```
export enum ErrorEnum {
    // Failure Messages
    INVALID_PASSWORD = "Invalid Password. Please try again with valid password.",
    INVALID_USER_ID = "Invalid User Id. Please try again with valid user id.",
    USER_ALREADY_EXISTS = "User Id Already Exists. Please try again with different User
Id.",
    NOT_AUTHENTICATED = "Please Login to access this URL.",
    NOT_AUTHORIZED = "You are not authorized to access this URL.",

    // Success Messages
    PASSWORD_RESET_SUCCESS = "Password Reset Success.",
    FOOD_DELETION_SUCCESS = "Food Item deleted successfully.",

    // Connection Messages
    JSON_CONNECTION_FAILED = "JSON Server Connection Failed.",
}
```

food-model.ts

```
export class FoodModel {
  public id?: number;

  constructor(
    public image: string,
    public foodName: string,
    public foodCategory: string,
    public price: number
  ) {}
}
```

user-model.ts

```
export class UserModel {
  public id?: number;
  public userRole?: string

  constructor(
    public userId: string,
    public password: string
  ) {}
}
```

Guards

admin-auth.guard.ts

```typescript
import { Injectable } from '@angular/core';
import { CanActivate, Router } from '@angular/router';
import { ErrorEnum } from '../model/error-enum';
import { AuthService } from '../services/auth.service';

@Injectable({
  providedIn: 'root'
})
export class AdminAuthGuard implements CanActivate {

  constructor(private authService:AuthService, private router : Router){  }

  canActivate() {
    if(this.authService.isAdmin()){
      return true;
    }
    alert(ErrorEnum.NOT_AUTHORIZED);
    this.router.navigate(['/api']);
    return false;
  }
}
```

authentication.guard.ts

```typescript
import { Injectable } from '@angular/core';
import { CanActivate, Router } from '@angular/router';
import { ErrorEnum } from '../model/error-enum';
import { AuthService } from '../services/auth.service';

@Injectable({
  providedIn: 'root'
})
export class AuthenticationGuard implements CanActivate {

  constructor(private authService:AuthService, private router : Router){  }

  canActivate() {
    if(this.authService.isUserLoggedIn()){
      return true;
    }
    alert(ErrorEnum.NOT_AUTHENTICATED);
    this.router.navigate(['/']);
    return false;
  }

}
```

user-auth.guard.ts

```
import { Injectable } from '@angular/core';
import { ActivatedRouteSnapshot, CanActivate, Router, RouterStateSnapshot, UrlTree } from '@angular/router';
import { Observable } from 'rxjs';
import { ErrorEnum } from '../model/error-enum';
import { AuthService } from '../services/auth.service';

@Injectable({
  providedIn: 'root'
})
export class UserAuthGuard implements CanActivate {

  constructor(private authService:AuthService, private router : Router){  }

  canActivate() {
    if(this.authService.isUser()){
      return true;
    }
    alert(ErrorEnum.NOT_AUTHORIZED);
    this.router.navigate(['/api']);
    return false;
  }
}
```

Environments

<<< Environment >>>

environment.ts

```
export const environment = {
  production: false,
  apiHost: 'http://localhost:3000'
};
```

environment.prod.ts

```
export const environment = {
  production: true
};
```

Components

<<< App Component >>>

app.component.html

```
<router-outlet></router-outlet>
```

app.component.ts

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

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'Kitchen Strory';
}
```

app.component.css

```
div {
  background-image: url("../assets/Background_Wallpaper.jpg");

  background-position: center;
  background-repeat: no-repeat;
  background-size: cover;
  position: relative;

  min-height: 85vh;
  min-width: 100%;
  margin: 0px;
  padding: 0px;
}
```

app.module.ts

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { HttpClientModule } from '@angular/common/http';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { AuthComponent } from './components/auth/auth.component';
import { SignInComponent } from './components/auth/sign-in/sign-in.component';
import { SignUpComponent } from './components/auth/sign-up/sign-up.component';
import { PasswordResetComponent } from './components/auth/password-reset/password-
reset.component';
import { PageNotFoundComponent } from './components/page-not-found/page-not-
found.component';
import { FormsModule, ReactiveFormsModule } from '@angular/forms';
import { FoodServiceService } from './services/food-service.service';
import { ApiComponent } from './components/api/api.component';
import { HeaderComponent } from './components/api/header/header.component';
import { SearchComponent } from './components/api/others/search/search.component';
import { FoodItemsComponent } from './components/api/others/food-items/food-
items.component';
import { OrderSummaryComponent } from './components/api/others/order-summary/order-
summary.component';
import { OrderPlacedComponent } from './components/api/others/order-placed/order-
placed.component';
import { AllFoodItemsComponent } from './components/api/others/all-food-items/all-food-
items.component';
import { AddNewFoodComponent } from './components/api/others/add-new-food/add-new-
food.component';
import { HomeComponent } from './components/api/others/home/home.component';
import { FooterComponent } from './components/api/footer/footer.component';

@NgModule({
  declarations: [
    AppComponent,
    HeaderComponent,
    AuthComponent,
    SignInComponent,
    SignUpComponent,
    PasswordResetComponent,
    SearchComponent,
    FoodItemsComponent,
    OrderSummaryComponent,
    OrderPlacedComponent,
    AllFoodItemsComponent,
    AddNewFoodComponent,
    HomeComponent,
    PageNotFoundComponent,
    FooterComponent,
    ApiComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule,
    ReactiveFormsModule,
    HttpClientModule
  ],
  providers: [FoodServiceService],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

app-routing.module.ts

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { ApiComponent } from './components/api/api.component';
import { AddNewFoodComponent } from './components/api/others/add-new-food/add-new-
food.component';
import { AllFoodItemsComponent } from './components/api/others/all-food-items/all-food-
items.component';
import { FoodItemsComponent } from './components/api/others/food-items/food-
items.component';
import { HomeComponent } from './components/api/others/home/home.component';
import { OrderPlacedComponent } from './components/api/others/order-placed/order-
placed.component';
import { OrderSummaryComponent } from './components/api/others/order-summary/order-
summary.component';
import { SearchComponent } from './components/api/others/search/search.component';
import { AuthComponent } from './components/auth/auth.component';
import { PasswordResetComponent } from './components/auth/password-reset/password-
reset.component';
import { SignInComponent } from './components/auth/sign-in/sign-in.component';
import { SignUpComponent } from './components/auth/sign-up/sign-up.component';
import { PageNotFoundComponent } from './components/page-not-found/page-not-
found.component';
import { AdminAuthGuard } from './guard/admin-auth.guard';
import { AuthenticationGuard } from './guard/authentication.guard';
import { UserAuthGuard } from './guard/user-auth.guard';

const routes: Routes = [
  {
    path: '',
    component: AuthComponent,
    children: [
      { path: '', component: SignInComponent },
      { path: 'sign-up', component: SignUpComponent },
      { path: 'log-out', component: AuthComponent },
      { path: 'password-reset', component: PasswordResetComponent },
    ],
  },
  {
    path: 'api',
    component: ApiComponent,
    children: [
      { path: '', component: HomeComponent, canActivate:[AuthenticationGuard]},
      { path: 'all-food-items', component: AllFoodItemsComponent,
canActivate:[AuthenticationGuard, AdminAuthGuard]},
      { path: 'add-new-food', component: AddNewFoodComponent,
canActivate:[AuthenticationGuard, AdminAuthGuard]},
      { path: 'search', component: SearchComponent, canActivate:[AuthenticationGuard,
UserAuthGuard]},
      { path: 'food-items', component: FoodItemsComponent,
canActivate:[AuthenticationGuard, UserAuthGuard] },
      { path: 'food-items/food-item/:id', component: OrderPlacedComponent,
canActivate:[AuthenticationGuard, UserAuthGuard] },
      { path: 'order-summary/:id', component: OrderSummaryComponent,
canActivate:[AuthenticationGuard, UserAuthGuard]},
    ],
    canActivate:[AuthenticationGuard]
  },
  {
    path: 'page-not-found',
    component: PageNotFoundComponent,
    data: { message: 'Page Not Found' },
  },
  { path: '**', redirectTo: 'page-not-found' },
];

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

<<< App/Components/Api/Header Component >>>

header.component.css

```css
.navbar-brand {
    background-color: white;
    padding: 0px;
}
nav {
    background-color: black;
}
a {
    font-size: medium;
}
```

header.component.html

```html
<div>
  <nav class="navbar navbar-expand-lg navbar-dark">
    <a
      routerLink="/api"
      [routerLinkActiveOptions]="{ exact: true }"
      routerLinkActive="active"
      class="navbar-brand nav-link"
    >
      <img
        src="./assets/KitchenStoryLogo.jpg"
        width="120px"
        height="80px"
        alt="Brand Logo"
        class="img-fluid img-responsive"
      />
    </a>
    <button
      class="navbar-toggler"
      type="button"
      data-toggle="collapse"
      data-target="#navbarNav"
      aria-controls="navbarNav"
      aria-expanded="false"
      aria-label="Toggle navigation"
    >
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarNav">
      <ul class="navbar-nav">
        <li class="nav-item" *ngIf="role === 'Admin'">
          <a
            routerLink="./all-food-items"
            routerLinkActive="active"
            class="nav-link h5"
            >All Food Items</a
          >
        </li>
        &nbsp;&nbsp;&nbsp;
        <li class="nav-item" *ngIf="role === 'Admin'">
          <a
            routerLink="./add-new-food"
            routerLinkActive="active"
            class="nav-link h5"
            >Add New Food</a
          >
        </li>
        &nbsp;&nbsp;&nbsp;
        <li class="nav-item" *ngIf="role === 'User'">
          <a routerLink="./search" routerLinkActive="active" class="nav-link h5"
            >Search Food</a
          >
        </li>
      </ul>
    </div>
    <div>
      <ul class="navbar-nav ml-auto">
        <li class="nav-item">
          <a routerLink="/" routerLinkActive="active" class="nav-link h5">
            <svg
              xmlns="http://www.w3.org/2000/svg"
              width="16"
              height="16"
              fill="currentColor"
              class="bi bi-box-arrow-right"
              viewBox="0 0 16 16"
            >
              <path
                fill-rule="evenodd"
                d="M10 12.5a.5.5 0 0 1-.5.5h-8a.5.5 0 0 1-.5-.5v-9a.5.5 0 0 1 .5-.5h8a.5.5
0 0 1 .5.5v2a.5.5 0 0 0 1 0v-2A1.5 1.5 0 0 0 9.5 2h-8A1.5 1.5 0 0 0 0 3.5v9A1.5 1.5 0 0 0
1.5 14h8a1.5 1.5 0 0 0 1.5-1.5v-2a.5.5 0 0 0-1 0v2z"
              />
              <path
                fill-rule="evenodd"
                d="M15.854 8.354a.5.5 0 0 0 0-.708l-3-3a.5.5 0 0 0-.708.708L14.293
7.5H5.5a.5.5 0 0 0 0 1h8.793l-2.147 2.146a.5.5 0 0 0 .708.708l3-3z"
              />
            </svg>
            Log Out
          </a>
        </li>
      </ul>
    </div>
  </nav>
</div>
```

header.component.ts

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

@Component({
  selector: 'app-header',
  templateUrl: './header.component.html',
  styleUrls: ['./header.component.css']
})
export class HeaderComponent implements OnInit {
role:string = "";
  constructor() { }

  ngOnInit(): void {
    this.role = sessionStorage.getItem('userRole');
  }

}
```

<<< App/Components/Api/Footer Component >>>

footer.component.css

```css
.footer {
    position: fixed;
    bottom: 0;
    width: 100%;
    height: 5%;
    line-height: 30px;
    background-color: black;
    padding: 0px 2%;
    text-align: right;
}
span {
    color: whitesmoke;
    font-family: cursive;
    font-size: larger;
    font-weight: bold;
}
```

footer.component.html

```html
<footer class="footer">
    <span> <i class="fa fa-copyright" aria-hidden="true">&copy;KitchenStory 2022</i>
</span>
</footer>
```

footer.component.ts

```ts
import { Component, OnInit } from '@angular/core';

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

  constructor() { }

  ngOnInit(): void {
  }

}
```

<<< App/Components/Api Component >>>

api.component.css

```css
div {
  background-image: url("/assets/Background_Wallpaper.jpg");

  background-position: center;
  background-repeat: no-repeat;
  background-size: cover;
  position: relative;

  min-height: 85vh;
  min-width: 100%;
  margin: 0px;
  padding: 0px;
}
```

api.component.html

```html
<div class="container-fluid" style="height: 100%">
  <app-header></app-header>
  <div class="container">
    <router-outlet></router-outlet>
  </div>
  <app-footer></app-footer>
</div>
```

api.component.ts

```typescript
import { Component, OnInit } from '@angular/core';

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

  constructor() { }

  ngOnInit(): void {
  }

}
```

<<< App/Components/Api/Others/Add New Food Component >>>

add-new-food.component.css

```css
.ng-touched .ng-invalid {
  border: 1px solid red;
}
.card {
  background-color: peru;
  margin: 2% 30%;
  border-radius: 2%;
  box-shadow: 0 5px 10px rgba(0, 0, 0, 0.2);
}
.card-header {
  text-align: center;
  font-weight: bolder;
  font-size: x-large;
  color: brown;
}
.card-body {
  padding: 2% 5% 1% 5%;
}
label {
  font-weight: bold;
}
div.form-group {
  margin-bottom: 5px;
}
button {
  margin-top: 5px;
  transition: 1s;
}
button:hover {
    background-color: #28a745!important;
    transform: scale(1.05);
}
span {
  padding: 2px;
}
```

add-new-food.component.html

```html
<div class="container-fluid">
  <div class="row">
    <div class="col-md-12">
      <div class="card">
        <div class="card-header">Add New Food Item</div>
        <div class="card-body">
          <form [formGroup]="foodDetailsForm" (ngSubmit)="onSubmit()">
            <div>
              <div class="form-group">
                <label>Food Name : </label>
                <input
                type="text"
                class="form-control"
                formControlName="foodName"
              />
              <div>
                <span
                  class="alert alert-danger"
                  *ngIf="
                    this.foodDetailsForm.get('foodName').invalid &&
                    this.foodDetailsForm.get('foodName').touched
                  "
                  >Please enter Food Name</span
                >
              </div>
            </div>

            <div class="form-group">
              <label>Food Category : </label>
              <select class="form-control" formControlName="foodCategory">
                <option value="">Select</option>
                <option value="Burger">Burger</option>
                <option value="Noodle">Noodle</option>
                <option value="Pancake">Pancake</option>
                <option value="Pasta">Pasta</option>
                <option value="Pizza">Pizza</option>
                <option value="Salad">Salad</option>
                <option value="Sandwich">Sandwich</option>
                <option value="Soup">Soup</option>
              </select>
              <div>
                <span
                  class="alert alert-danger"
                  *ngIf="
                    this.foodDetailsForm.get('foodCategory').invalid &&
                    this.foodDetailsForm.get('foodCategory').touched
                  "
                  >Please enter Food Category</span
                >
              </div>
            </div>

            <div class="form-group">
              <label>Price : </label>
              <input
                type="number"
                class="form-control"
                formControlName="price"
              />
              <div>
                <span
                  class="alert alert-danger"
                  *ngIf="
                    this.foodDetailsForm.get('price').invalid &&
                    this.foodDetailsForm.get('price').touched
                  "
                  >Please enter Price</span
                >
              </div>
            </div>

            <div class="form-group">
              <label>Image URL : </label>
              <input
                type="text"
                class="form-control"
                formControlName="imageUrl"
              />
              <div>
                <span
                  class="alert alert-danger"
                  *ngIf="
                    this.foodDetailsForm.get('imageUrl').invalid &&
                    this.foodDetailsForm.get('imageUrl').touched
                  "
                  >Please enter Image URL</span
                >
              </div>
            </div>

            <div class="form-group text-center">
              <button
                class="btn btn-primary"
                [disabled]="foodDetailsForm.invalid"
              >
                Add Food Item
              </button>
            </div>
          </div>
        </form>
      </div>
    </div>
  </div>
</div>
</div>
```

add-new-food.component.ts

```
import { Component, OnInit } from '@angular/core';
import { FormControl, FormGroup, Validators } from '@angular/forms';
import { Router } from '@angular/router';
import { ErrorEnum } from 'src/app/model/error-enum';
import { FoodModel } from 'src/app/model/food-model';
import { FoodServiceService } from 'src/app/services/food-service.service';

@Component({
  selector: 'app-add-new-food',
  templateUrl: './add-new-food.component.html',
  styleUrls: ['./add-new-food.component.css'],
})
export class AddNewFoodComponent implements OnInit {
  foodDetailsForm!: FormGroup;

  constructor(
    private foodService: FoodServiceService,
    private router: Router
  ) {}

  ngOnInit(): void {
    this.foodDetailsForm = new FormGroup({
      foodName: new FormControl('', Validators.required),
      foodCategory: new FormControl('', Validators.required),
      price: new FormControl('', [Validators.required, Validators.min(0)]),
      imageUrl: new FormControl('./assets/', Validators.required),
    });
  }

  onSubmit() {
    let foodItem = new FoodModel(
      this.foodDetailsForm.value.imageUrl,
      this.foodDetailsForm.value.foodName,
      this.foodDetailsForm.value.foodCategory,
      this.foodDetailsForm.value.price
    );
    console.log(foodItem);
    this.foodService.addFoodItem(foodItem).subscribe(
      (res) => {
        this.router.navigate(['/api']);
      },
      (error) => {
        alert(ErrorEnum.JSON_CONNECTION_FAILED);
      }
    );
  }
}
```

<<< App/Components/Api/Others/ All Food Items Component >>>

all-food-items.component.css

```css
.no-products-msg {
  text-align: center;
  font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;
  font-size: xx-large;
  font-weight: bold;
  margin: 2% 15%;
  background-color: darkgrey;
}div p {
  margin-bottom: 0px;
}
table {
  background-color: thistle;
  margin-bottom: 1%;
}
tr {
  text-align: center;
  color: darkslategray;
}
th {
  padding: 10px 2px;
}
td {
  padding: 2px;
  vertical-align: middle;
}
div.col-md-12 {
  padding: 1% 10% 3%;
}
/* Table Fields Styling */
.srNo-styling {
  width: 5%;
}
.image-styling {
  width: 20%;
}
.foodName-styling {
  width: 30%;
}
.category-styling {
  width: 15%;
}
.price-styling {
  width: 10%;
}
.qty-styling {
  width: 10%;
}
.purchase-styling {
  width: 25%;
}
```

```
img {
  width: 90%;
  height: 60%;
}
.td-image-styling {
  padding: 2px 0px;
  transition: 1s;
}
.td-image-styling:hover {
  transform: scale(1.1);
}
button:hover {
  background-color: #28a745;
}
a {
  background: none;
  border: 0px;
  padding: 0px;
}
td button:hover {
  background-color: #dc3545 !important;
}
```

all-food-items.component.html

```
<div class="container-fluid">
  <div class="row">
    <div class="col-md-12">
      <div class="no-products-msg text-danger" *ngIf="isListEmpty">
        <p>No Food Items Found</p>
        <p style="font-size: large">
          (Please search with valid product categories as mentioned on search
          page)
        </p>
      </div>

      <div *ngIf="!isListEmpty">
        <table class="table table-hover">
          <thead>
            <tr>
              <th class="srNo-styling">Sr. No.</th>
              <th class="image-styling">Image</th>
              <th class="foodName-styling">Food Name</th>
              <th class="category-styling">Food Category</th>
              <th class="price-styling">Price</th>
              <th class="purchase-styling">Purchase</th>
            </tr>
          </thead>
          <tbody>
            <tr *ngFor="let food of allFoodItems; let i = index">
              <td>{{ i + 1 }}</td>
              <td class="td-image-styling">
```

```
                <img
                  [src]="food.image"
                  [alt]="'{{' + food.image + '}} Image Not Found'"
                  width="30%"
                  height="10%"
                />
              </td>
              <td>{{ food.foodName }}</td>
              <td>{{ food.foodCategory }}</td>
              <td>{{ food.price | currency: "INR" }}</td>
              <td>
                <button class="btn btn-primary" (click)="deleteFood(food.id)">
                  Delete
                </button>
              </td>
            </tr>
          </tbody>
        </table>
      </div>
    </div>
  </div>
</div>
```

all-food-items.component.ts

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';
import { ErrorEnum } from 'src/app/model/error-enum';
import { FoodModel } from 'src/app/model/food-model';
import { FoodServiceService } from 'src/app/services/food-service.service';

@Component({
  selector: 'app-all-food-items',
  templateUrl: './all-food-items.component.html',
  styleUrls: ['./all-food-items.component.css'],
})
export class AllFoodItemsComponent implements OnInit {
  allFoodItems: FoodModel[];
  isListEmpty: Boolean = false;

  constructor(
    private foodService: FoodServiceService
  ) {}

  ngOnInit(): void {
    this.fetchAllFoodItems();
  }

  fetchAllFoodItems() {
    this.foodService.getAllFoodItems().subscribe(
      (res) => {
        this.allFoodItems = res;
        if (this.allFoodItems.length === 0) {
          this.isListEmpty = true;
        } else {
          this.isListEmpty = false;
        }
      },
      (error) => {
        alert(ErrorEnum.JSON_CONNECTION_FAILED);
      }
    );
  }

  deleteFood(id: number) {
    this.foodService.deleteFoodItem(id).subscribe(
      (res) => {
        alert(ErrorEnum.FOOD_DELETION_SUCCESS);
      },
      (error) => {
        alert(ErrorEnum.JSON_CONNECTION_FAILED);
      }
    );
    this.refresh();
  }

  refresh(): void {
    window.location.reload();
  }
}
```

<<< App/Components/Api/Others/ Food Items Component >>>

food-items.component.css

```css
.no-products-msg {
  text-align: center;
  font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;
  font-size: xx-large;
  font-weight: bold;
  margin: 2% 15%;
  background-color: darkgrey;
}
div p {
  margin-bottom: 0px;
}
table {
  background-color: thistle;
  margin-bottom: 1%;
}
tr {
  text-align: center;
  color: darkslategray;
}
th {
  padding: 10px 2px;
}
td {
  padding: 2px;
  vertical-align: middle;
}
div.col-md-12 {
  padding: 1% 10% 3%;
}
/* Table Fields Styling */
.srNo-styling {
  width: 5%;
}
.image-styling {
  width: 20%;
}
.foodName-styling {
  width: 30%;
}
.category-styling {
  width: 15%;
}
.price-styling {
  width: 10%;
}
.qty-styling {
  width: 10%;
}
.purchase-styling {
  width: 25%;
}
img {
  width: 90%;
  height: 60%;
}
.td-image-styling {
  padding: 2px 0px;
  transition: 1s;
}
.td-image-styling:hover {
  transform: scale(1.1);
}
button:hover {
  background-color: #28a745 !important;
}
a {
  background: none;
  border: 0px;
  padding: 0px;
}
```

food-items.component.html

```html
<div class="container-fluid">
  <div class="row">
    <div class="col-md-12">
      <div class="no-products-msg text-danger" *ngIf="isCtgryListEmpty">
        <p>No Food Items Found</p>
        <p style="font-size: large">
          (No food items available with the food category provided)
        </p>
        <p style="font-size: large; color: #28a745 !important">
          (Try to search food in another category)
        </p>
      </div>
      <table class="table table-hover" *ngIf="!isCtgryListEmpty">
        <thead>
          <tr>
            <th class="srNo-styling">Sr. No.</th>
            <th class="image-styling">Image</th>
            <th class="foodName-styling">Food Name</th>
            <th class="category-styling">Food Category</th>
            <th class="price-styling">Price</th>
            <th class="purchase-styling">Purchase</th>
          </tr>
        </thead>
        <tbody>
          <tr *ngFor="let food of filteredFoodItems; let i = index">
            <td>{{ i + 1 }}</td>
            <td class="td-image-styling">
              <img
                [src]="food.image"
                alt="Image Not Found"
                width="30%"
                height="10%"
              />
            </td>
            <td>{{ food.foodName }}</td>
            <td>{{ food.foodCategory }}</td>
            <td>{{ food.price | currency: "INR" }}</td>
            <td>
              <a [routerLink]="['food-item', food.id]" class="btn btn-primary">
                <button class="btn btn-primary">Buy Now</button>
              </a>
            </td>
          </tr>
        </tbody>
      </table>
    </div>
  </div>
</div>
```

food-items.component.ts

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';
import { ErrorEnum } from 'src/app/model/error-enum';
import { FoodModel } from 'src/app/model/food-model';
import { FoodServiceService } from 'src/app/services/food-service.service';

@Component({
  selector: 'app-food-items',
  templateUrl: './food-items.component.html',
  styleUrls: ['./food-items.component.css'],
})
export class FoodItemsComponent implements OnInit {
  filteredFoodItems: FoodModel[];
  isCtgryListEmpty:Boolean = false;

  constructor(
    private foodService: FoodServiceService,
    private activatedRoute: ActivatedRoute
  ) {}

  ngOnInit(): void {
    let category: string;
    category = this.activatedRoute.snapshot.queryParamMap.get('category');
    this.foodService.getFoodsByCategory(category).subscribe(
      (res)=>{
        this.filteredFoodItems = res;
        if (this.filteredFoodItems.length === 0) {
          this.isCtgryListEmpty = true;
        } else {
          this.isCtgryListEmpty = false;
        }
      },(error)=>{
        alert(ErrorEnum.JSON_CONNECTION_FAILED);
      }
    );
  }
}
```

<<< App/Components/Api/Others/ Home Component >>>

home.component.css

```
div {
  padding-top: 15%;
  min-width: 100%;
}
marquee {
  font-family: 'Trebuchet MS', 'Lucida Sans Unicode', 'Lucida Grande', 'Lucida Sans',
Arial, sans-serif;
  font-size: xx-large;
  font-weight: bolder;
  color: rgb(67, 19, 19);
  min-height: 70%;
}
p {
    text-align: center;
    margin: 0% 40%;;
    background-color: thistle;
}
```

home.component.html

```
<div>
  <marquee behavior="slide" direction="down"> <p>Welcome  to  Kitchen  Story</p>
</marquee>
</div>
```

home.component.ts

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

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

  constructor() { }

  ngOnInit(): void {
  }

}
```

<<< App/Components/Api/Others/ Order Placed Component >>>

order-placed.component.css

```css
.container-fluid {
  background: none;
}
div.col-md-12 {
  padding: 0px;
}
div.card {
  margin: 1% 20% 0%;
  background-color: darkgray;
  border-radius: 5%;
}
div.card-header {
  background-color: grey;
  padding: 0px;
}
.card-title {
  font-size: xx-large;
  font-family: fantasy;
  font-weight: bolder;
  text-align: center;
  margin-bottom: 0px;
}
div.card-body {
  padding: 0px;
}
table {
  margin-bottom: 0px;
}
td {
  padding: 5px;
  width: 60%;
}
.row-title {
  font-weight: bold;
  width: 40%;
}
.td-image-styling {
  text-align: center;
  transition: 1s;
}
td.btn-styling {
  text-align: center;
  transition: 1s;
  }
.td-image-styling:hover{
  transform: scale(1.1);
}
.btn-primary:hover {
  transform: scale(1.1);
  background-color: #28a745;
}
```

order-placed.component.html

```
<div class="container-fluid" *ngIf="renderPage">
  <div class="card">
    <div class="card-header">
      <p class="card-title">Food Details</p>
    </div>
    <div class="card-body">
      <table class="table table-hover">
        <tbody>
          <tr>
            <td colspan="2" class="td-image-styling">
              <img
                [src]="foodItem.image"
                alt="Image Not Found"
                width="25%"
                height="8%"
              />
            </td>
          </tr>
          <tr>
            <td class="row-title">Food Id :</td>
            <td>{{ foodItem.id }}</td>
          </tr>

          <tr>
            <td class="row-title">Food Name :</td>
            <td>{{ foodItem.foodName }}</td>
          </tr>
          <tr>
            <td class="row-title">Food Category :</td>
            <td>{{ foodItem.foodCategory }}</td>
          </tr>
          <tr>
            <td class="row-title">Food Price :</td>
            <td>{{ foodItem.price | currency: "INR" }}</td>
          </tr>
          <tr>
            <td colspan="2" class="btn-styling">
              <button
                class="btn btn-primary"
                [routerLink]="['/api/order-summary', foodItem.id]"
              >
                <svg
                  xmlns="http://www.w3.org/2000/svg"
                  width="16"
                  height="16"
                  fill="currentColor"
                  class="bi bi-credit-card"
                  viewBox="0 0 16 16"
                >
                  <path
                    d="M0 4a2 2 0 0 1 2-2h12a2 2 0 0 1 2 2v8a2 2 0 0 1-2 2H2a2 2 0 0 1-2-
2V4zm2-1a1 1 0 0 0-1 1v1h14V4a1 1 0 0 0-1-1H2zm13 4H1v5a1 1 0 0 0 1 1h12a1 1 0 0 0 1-1V7z"
                  />
                  <path
                    d="M2 10a1 1 0 0 1 1-1h1a1 1 0 0 1 1 1v1a1 1 0 0 1-1 1H3a1 1 0 0 1-1-
1v-1z"
                  />
                </svg>
                Pay & Place Order
              </button>
            </td>
          </tr>
        </tbody>
      </table>
    </div>
  </div>
</div>
```

order-placed.component.ts

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Params, Router } from '@angular/router';
import { ErrorEnum } from 'src/app/model/error-enum';
import { FoodModel } from 'src/app/model/food-model';
import { FoodServiceService } from 'src/app/services/food-service.service';

@Component({
  selector: 'app-order-placed',
  templateUrl: './order-placed.component.html',
  styleUrls: ['./order-placed.component.css'],
})
export class OrderPlacedComponent implements OnInit {
  foodItem: FoodModel;
  renderPage:Boolean = false;

  constructor(
    private activatedRoute: ActivatedRoute,
    private foodService: FoodServiceService
  ) {}

  ngOnInit(): void {
    this.activatedRoute.params.subscribe((params: Params) => {
      let foodId = +params['id'];
      this.foodService.getFoodById(foodId).subscribe(
        (res) => {
          this.foodItem = res;
          this.renderPage = true;
        },
        (error) => {
          alert(ErrorEnum.JSON_CONNECTION_FAILED);
        }
      );
    });
  }

}
```

<<< App/Components/Api/Others/ Order Summary Component >>>

order-summary.component.css

```css
.container-fluid {
  background: none;
  padding: 1% 15% 1%;
}
div.col-md-12 {
  padding: 0px;
}
div.card {
  padding: 1% 20% 0%;
  background: none;
}
div.card-header {
  padding: 0px;
  background-color: peru;
}
div.card-body {
  background-color: burlywood;
}
p {
  text-align: center;
  font-family: Franklin Gothic Medium, "Arial Narrow", Arial, sans-serif;
  font-size: larger;
}
.card-title {
  font-size: x-large;
  font-family: "Lucida Sans", "Lucida Sans Regular", "Lucida Grande",
    "Lucida Sans Unicode", Geneva, Verdana, sans-serif;
  font-weight: bolder;
  text-align: center;
  margin-bottom: 0px;
}
div.card-body {
  padding: 0px;
}
table {
  margin-bottom: 0px;
  border-color: black;
}
td {
  padding: 5px;
  width: 60%;
  vertical-align: middle;
}
.greetMsg {
  margin: 10px;
}
.row-title {
  width: 20%;
  font-weight: bold;
}
```

```css
.td-image-styling {
  text-align: center;
  width: 50%;
  transition: 1s;
}
.btn-styling {
  text-align: center;
  transition: 1s;
}
.td-image-styling:hover {
  transform: scale(1.1);
}
.btn-primary:hover {
  transform: scale(1.1);
  background-color: #28a745;
}
```

<u>order-summary.component.html</u>

```html
<div class="container-fluid" *ngIf="renderPage">
  <div class="card">
    <div class="card-header">
      <p class="card-title">Wow...!!! Order Placed successfully...!!!</p>
      <p>Your Order Id is <b>{{orderId}}</b> </p>
    </div>
    <div class="card-body">
      <table class="table table-striped table-bordered table-hover">
        <tbody>
          <tr>
            <td rowspan="4" class="td-image-styling">
              <img
                [src]="foodItem.image"
                alt="Image Not Found"
                width="90%"
                height="70%"
              />
            </td>
            <td class="row-title">Food Name</td>
            <td>{{ foodItem.foodName }}</td>
          </tr>
          <tr>
            <td class="row-title">Food Category</td>
            <td>{{ foodItem.foodCategory }}</td>
          </tr>
          <tr>
            <td class="row-title">Food Price</td>
            <td>{{ foodItem.price | currency: "INR" }}</td>
          </tr>
          <tr>
            <td class="row-title">{{foodItem.foodCategory}} Quantity</td>
            <td>1</td>
          </tr>
          <tr>
            <td colspan="3">
                <p class="greetMsg">
                    Thanks for Visiting us...
                    <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16"
fill="currentColor" class="bi bi-emoji-heart-eyes-fill" viewBox="0 0 16 16">
                        <path d="M8 0a8 8 0 1 0 0 16A8 8 0 0 0 8 0zM4.756 4.566c.763-1.424
4.02-.12.952 3.434-4.496-1.596-2.35-4.298-.952-3.434zm6.559 5.448a.5.5 0 0 1
.548.736A4.498 4.498 0 0 1 7.965 13a4.498 4.498 0 0 1-3.898-2.25.5.5 0 0 1 .548-
.736h.005l.017.005.067.015.252.055c.215.046.515.108.857.169.693.124 1.522.242 2.152.242.63
0 1.46-.118 2.152-.242a26.58 26.58 0 0 0 1.109-.224l.067-.015.017-.004.005-.002zm-.07-
5.448c1.397-.864 3.543 1.838-.953 3.434-3.067-3.554.19-4.858.952-3.434z"/>
                    </svg>&nbsp;
                    <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16"
fill="currentColor" class="bi bi-emoji-heart-eyes-fill" viewBox="0 0 16 16">
                        <path d="M8 0a8 8 0 1 0 0 16A8 8 0 0 0 8 0zM4.756 4.566c.763-1.424
4.02-.12.952 3.434-4.496-1.596-2.35-4.298-.952-3.434zm6.559 5.448a.5.5 0 0 1
.548.736A4.498 4.498 0 0 1 7.965 13a4.498 4.498 0 0 1-3.898-2.25.5.5 0 0 1 .548-
.736h.005l.017.005.067.015.252.055c.215.046.515.108.857.169.693.124 1.522.242 2.152.242.63
0 1.46-.118 2.152-.242a26.58 26.58 0 0 0 1.109-.224l.067-.015.017-.004.005-.002zm-.07-
5.448c1.397-.864 3.543 1.838-.953 3.434-3.067-3.554.19-4.858.952-3.434z"/>
                    </svg>&nbsp;
                    <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16"
fill="currentColor" class="bi bi-emoji-heart-eyes-fill" viewBox="0 0 16 16">
                        <path d="M8 0a8 8 0 1 0 0 16A8 8 0 0 0 8 0zM4.756 4.566c.763-1.424
4.02-.12.952 3.434-4.496-1.596-2.35-4.298-.952-3.434zm6.559 5.448a.5.5 0 0 1
.548.736A4.498 4.498 0 0 1 7.965 13a4.498 4.498 0 0 1-3.898-2.25.5.5 0 0 1 .548-
.736h.005l.017.005.067.015.252.055c.215.046.515.108.857.169.693.124 1.522.242 2.152.242.63
0 1.46-.118 2.152-.242a26.58 26.58 0 0 0 1.109-.224l.067-.015.017-.004.005-.002zm-.07-
5.448c1.397-.864 3.543 1.838-.953 3.434-3.067-3.554.19-4.858.952-3.434z"/>
                    </svg>&nbsp;
                </p>
                <div class="btn-styling">
                <button [routerLink]="['/api']" class="btn btn-primary">
                    Go Home
                </button>
            </div>
            </td>
          </tr>
        </tbody>
      </table>
    </div>
  </div>
</div>
```

order-summary.component.ts

```typescript
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Params } from '@angular/router';
import { ErrorEnum } from 'src/app/model/error-enum';
import { FoodModel } from 'src/app/model/food-model';
import { FoodServiceService } from 'src/app/services/food-service.service';
import * as uuid from 'uuid';

@Component({
  selector: 'app-order-summary',
  templateUrl: './order-summary.component.html',
  styleUrls: ['./order-summary.component.css'],
})
export class OrderSummaryComponent implements OnInit {
  foodItem: FoodModel;
  orderId: string;
  renderPage:Boolean = false;

  constructor(
    private activatedRoute: ActivatedRoute,
    private foodService: FoodServiceService
  ) {}

  ngOnInit(): void {
    this.activatedRoute.params.subscribe((params: Params) => {
      const foodId = +params['id'];
      this.foodService.getFoodById(foodId).subscribe(
        (res) => {
          this.foodItem = res;
          this.orderId = this.generateOrderId();
          this.renderPage = true;
        },
        (error) => {
          alert(ErrorEnum.JSON_CONNECTION_FAILED);
        }
      );
    });
  }

  generateOrderId() {
    return uuid.v4();
  }
}
```

<<< App/Components/Api/Others/ Search Component >>>

search.component.css

```css
.ng-touched .ng-invalid {
  border: 1px solid red;
}
.card {
  background-color: peru;
  margin: 2% 30%;
  border-radius: 2%;
  box-shadow: 0 5px 10px rgba(0, 0, 0, 0.2);
}
.card-header {
  text-align: center;
  font-weight: bolder;
  font-size: x-large;
  color: brown;
}
.card-body {
  padding: 2% 5% 1% 5%;
}
label {
  font-weight: bold;
}
button {
  margin-top: 10px;
  transition: 1s;
}
button:hover {
  background-color: #28a745 !important;
  transform: scale(1.05);
}
span {
padding: 0px;
}
```

search.component.html

```html
<div class="container-fluid">
  <div class="row">
    <div class="col-md-12">
      <div class="card">
        <div class="card-header">Search Food Items</div>
        <div class="card-body">
          <form [formGroup]="searchForm" (ngSubmit)="onSubmit()">
            <div>
              <div class="form-group">
                <label> Please Select Food Category </label>
                <select class="form-control" formControlName="foodCategory">
                  <option value="">Select</option>
                  <option value="Burger">Burger</option>
                  <option value="Noodle">Noodle</option>
                  <option value="Pancake">Pancake</option>
                  <option value="Pasta">Pasta</option>
                  <option value="Pizza">Pizza</option>
                  <option value="Salad">Salad</option>
                  <option value="Sandwich">Sandwich</option>
                  <option value="Soup">Soup</option>
                </select>
                <div>
                  <span
                    class="alert alert-danger"
                    *ngIf="
                      this.searchForm.get('foodCategory').invalid &&
                      this.searchForm.get('foodCategory').touched
                    "
                    >Please Select Food Category</span
                  >
                </div>

                <div class="form-group text-center">
                  <button
                    class="btn btn-success"
                    [disabled]="searchForm.invalid"
                  >
                    Find Food for me
                  </button>
                </div>
              </div>
            </div>
          </form>
        </div>
      </div>
    </div>
  </div>
</div>
```

search.component.ts

```
import { Component, OnInit } from '@angular/core';
import { FormControl, FormGroup, Validators } from '@angular/forms';
import { Router } from '@angular/router';
import { FoodModel } from 'src/app/model/food-model';

@Component({
  selector: 'app-search',
  templateUrl: './search.component.html',
  styleUrls: ['./search.component.css'],
})
export class SearchComponent implements OnInit {
  searchForm!: FormGroup;
  foodItems: FoodModel[];
  foodCategory:string;
  constructor(
    private router: Router
  ) {}

  ngOnInit(): void {
    this.searchForm = new FormGroup({
      foodCategory: new FormControl('', Validators.required),
    });
  }

  onSubmit() {
    this.foodCategory = this.searchForm.value.foodCategory;
    this.router.navigate(['/api/food-items'],{queryParams:{category: this.foodCategory}});
  }
}
```

<<< App/Components/Api/Auth Component >>>

auth.component.css

```css
div.auth-page {
    background-image: url('/assets/Background_Wallpaper.jpg');

    background-position: center;
    background-repeat: no-repeat;
    background-size: cover;
    position: relative;

    min-height: 100vh;
    min-width: 100%;
    margin: 0px;
    padding: 0px;
}
```

auth.component.html

```html
<div class="auth-page">
  <router-outlet></router-outlet>
</div>
```

auth.component.ts

```typescript
import { Component, OnInit } from '@angular/core';
import { FormControl, FormGroup, Validators } from '@angular/forms';
import { Router } from '@angular/router';
import { UserModel } from 'src/app/model/user-model';
import { UserService } from 'src/app/services/user.service';

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

  constructor() { }

  ngOnInit(): void { }
}
```

<<< App/Components/Api/Auth/ Password Reset Component >>>

password-reset.component.css

```
.password-reset-form {
  padding: 10% 35% 0%;
}
.ng-touched .ng-invalid {
  border: 1px solid red;
}
.card {
  background-color: peru;
  border-radius: 3%;
  box-shadow: 0 5px 10px rgba(0, 0, 0, 0.2);
}
.card-header {
  text-align: center;
  font-weight: bolder;
  font-size: x-large;
  color: brown;
}
.card-body {
  padding: 2% 5% 1% 5%;
}
label {
  font-weight: bold;
}
div.form-group {
  margin-bottom: 5px;
}
button {
  margin-top: 5px;
  transition: 1s;
}
button:hover {
  background-color: #28a745 !important;
  transform: scale(1.1);
}
span {
  padding: 2px;
}
```

password-reset.component.html

```
<div class="password-reset-form">
  <div class="card">
    <div class="card-header">Password Reset</div>
    <div class="card-body">
      <form [formGroup]="passwordResetForm" (ngSubmit)="onSubmit()">
        <div>
          <div class="form-group">
            <label>User Id : </label>
            <input type="text" class="form-control" formControlName="userId" />
            <div>
              <span
                class="alert alert-danger"
                *ngIf="
                  this.passwordResetForm.get('userId').invalid &&
                  this.passwordResetForm.get('userId').touched
                "
                >Please enter user id</span
              >
            </div>
          </div>

          <div class="form-group">
            <label> New Password : </label>
            <input
              type="password"
              class="form-control"
              formControlName="password"
            />
            <div>
              <span
                class="alert alert-danger"
                *ngIf="
                  this.passwordResetForm.get('password').invalid &&
                  this.passwordResetForm.get('password').touched
                "
                >Password should be minimum 8 and maximum 16 characters
                long</span
              >
            </div>
          </div>
          <br />
          <div class="form-group text-center">
            <button class="btn btn-primary" [routerLink]="['/']">Login</button
            >&nbsp;
            <button
              class="btn btn-primary"
              [disabled]="passwordResetForm.invalid"
            >
              Password Reset
            </button>
          </div>
        </div>
      </form>
    </div>
  </div>
</div>
```

password-reset.component.ts

```
import { Component, OnInit } from '@angular/core';
import { FormControl, FormGroup, Validators } from '@angular/forms';
import { Router } from '@angular/router';
import { ErrorEnum } from 'src/app/model/error-enum';
import { UserModel } from 'src/app/model/user-model';
import { UserService } from 'src/app/services/user.service';

@Component({
  selector: 'app-password-reset',
  templateUrl: './password-reset.component.html',
  styleUrls: ['./password-reset.component.css'],
})
export class PasswordResetComponent implements OnInit {
  passwordResetForm!: FormGroup;
  userResp: any[];

  constructor(private userService: UserService, private router: Router) {}

  ngOnInit(): void {
    sessionStorage.removeItem('userId');
    sessionStorage.removeItem('password');
    sessionStorage.removeItem('userRole');

    this.passwordResetForm = new FormGroup({
      userId: new FormControl('', Validators.required),
      password: new FormControl('', [
        Validators.required,
        Validators.minLength(5),
        Validators.maxLength(16),
      ]),
    });
  }

  onSubmit() {
    let userModel = new UserModel(
      this.passwordResetForm.value.userId,
      this.passwordResetForm.value.password
    );
    this.userService.signingInUser(userModel).subscribe(
      (res) => {
        this.userResp = res;
        if (this.userResp.length != 0) {
          userModel.id = this.userResp[0].id;
          userModel.userRole = this.userResp[0].userRole;

          this.userService.resetPassowrd(userModel).subscribe(
            (res) => {
              alert(ErrorEnum.PASSWORD_RESET_SUCCESS);
              this.router.navigate(['/']);
            },
            (error) => {
              alert(ErrorEnum.JSON_CONNECTION_FAILED);
            }
          );
        } else {
          this.passwordResetForm.reset();
          alert(ErrorEnum.INVALID_USER_ID);
          return;
        }
      },
      (error) => {
        alert(ErrorEnum.JSON_CONNECTION_FAILED);
      }
    );
  }

}
```

<<< App/Components/Api/Auth/ Sign In Component >>>

sign-in.component.css

```css
.sign-in-form {
  padding: 10% 35% 0%;
}
.ng-touched .ng-invalid {
  border: 1px solid red;
}
.card {
  background-color: peru;
  border-radius: 3%;
  box-shadow: 0 5px 10px rgba(0, 0, 0, 0.2);
}
.card-header {
  text-align: center;
  font-weight: bolder;
  font-size: x-large;
  color: brown;
}
.card-body {
  padding: 2% 5% 1% 5%;
}
label {
  font-weight: bold;
}
div.form-group {
  margin-bottom: 5px;
}
button {
  margin-top: 5px;
  transition: 1s;
}
button:hover {
  background-color: #28a745 !important;
  transform: scale(1.1);
}
span {
  padding: 2px;
}
span.new-user-link{
  min-width: 50%;
  text-align: left;
}
span.pwd-rst-link{
  min-width: 50%;
  text-align: right;
}
a{
  color: oldlace;
}
a:hover{
  color: blue;
}
```

sign-in.component.html

```
<div class="sign-in-form">
  <div class="card">
    <div class="card-header">Login</div>
    <div class="card-body">
      <form [formGroup]="SignInForm" (ngSubmit)="onSubmit()">
        <div>
          <div class="form-group">
            <label>User Id : </label>
            <input type="text" class="form-control" formControlName="userId" />
            <div>
              <span
                class="alert alert-danger"
                *ngIf="
                  this.SignInForm.get('userId').invalid &&
                  this.SignInForm.get('userId').touched
                "
                >Please enter user id</span
              >
            </div>
          </div>

          <div class="form-group">
            <label>Password : </label>
            <input
              type="password"
              class="form-control"
              formControlName="password"
            />
            <div>
              <span
                class="alert alert-danger"
                *ngIf="
                  this.SignInForm.get('password').invalid &&
                  this.SignInForm.get('password').touched
                "
                >Password should be minimum 8 and maximum 16 characters
                long</span
              >
            </div>
          </div>
          <div class="form-group">
            <div class="pwd-rst-link">
              <a [routerLink]="['password-reset']">Forgot password?</a>
            </div>
            <div class="new-user-link">
              Don't have an account?<a [routerLink]="['sign-up']"
                >Register here</a
              >
            </div>
          </div>
          <br />
          <div class="form-group text-center">
            <button
              type="submit"
              class="btn btn-primary"
              [disabled]="SignInForm.invalid"
            >
              Sign In
            </button>
          </div>
        </div>
      </form>
    </div>
  </div>
</div>
```

sign-in.component.ts

```
import { Component, OnInit } from '@angular/core';
import { FormControl, FormGroup, Validators } from '@angular/forms';
import { Router } from '@angular/router';
import { ErrorEnum } from 'src/app/model/error-enum';
import { UserModel } from 'src/app/model/user-model';
import { UserService } from 'src/app/services/user.service';

@Component({
  selector: 'app-sign-in',
  templateUrl: './sign-in.component.html',
  styleUrls: ['./sign-in.component.css'],
})
export class SignInComponent implements OnInit {
  SignInForm!: FormGroup;
  userResp: any[];

  constructor(private userService: UserService, private router: Router) {}

  ngOnInit(): void {
    sessionStorage.removeItem('userId');
    sessionStorage.removeItem('password');
    sessionStorage.removeItem('userRole');

    this.SignInForm = new FormGroup({
      userId: new FormControl('', Validators.required),
      password: new FormControl('', [
        Validators.required,
        Validators.minLength(5),
        Validators.maxLength(16)
      ]),
    });
  }
  onSubmit() {
    let userModel = new UserModel(
      this.SignInForm.value.userId,
      this.SignInForm.value.password
    );
    this.userService.signingInUser(userModel).subscribe(
      (res) => {
        this.userResp = res;
        if (this.userResp.length != 0) {
          if (this.userResp[0].password !== userModel.password) {
            this.SignInForm.reset();
            alert(ErrorEnum.INVALID_PASSWORD);
            return;
          }

          sessionStorage.setItem('userId', this.userResp[0].userId);
          sessionStorage.setItem('password', this.userResp[0].password);
          sessionStorage.setItem('userRole', this.userResp[0].userRole);

          // alert(ErrorEnum.SIGN_IN_SUCCESS);
          this.router.navigate(['/api']);
        } else {
          this.SignInForm.reset();
          alert(ErrorEnum.INVALID_USER_ID);
          return;
        }
      },
      (error) => {
        alert(ErrorEnum.JSON_CONNECTION_FAILED);
      }
    );
  }
}
```

<<< App/Components/Api/Auth/ Sign Up Component >>>

sign-up.component.css

```css
.ng-touched .ng-invalid {
  border: 1px solid red;
}
.sign-up-form {
  padding: 10% 35% 0%;
}
.card {
  background-color: peru;
  border-radius: 3%;
  box-shadow: 0 5px 10px rgba(0, 0, 0, 0.2);
}
.card-header {
  text-align: center;
  font-weight: bolder;
  font-size: x-large;
  color: brown;
}
.card-body {
  padding: 2% 5% 1% 5%;
}
label {
  font-weight: bold;
}
div.form-group {
  margin-bottom: 5px;
}
button {
  margin-top: 5px;
  transition: 1s;
}
button:hover {
  background-color: #28a745 !important;
  transform: scale(1.1);
}
span {
  padding: 2px;
}
```

sign-up.component.html

```html
<div class="sign-up-form">
  <div class="card">
    <div class="card-header">Sign Up</div>
    <div class="card-body">
      <form [formGroup]="SignUpForm" (ngSubmit)="onSubmit()">
        <div>
          <div class="form-group">
            <label>User Id : </label>
            <input type="text" class="form-control" formControlName="userId" />
            <div>
              <span
                class="alert alert-danger"
                *ngIf="
                  this.SignUpForm.get('userId').invalid &&
                  this.SignUpForm.get('userId').touched
                "
                >Please enter user id</span
              >
            </div>
          </div>

          <div class="form-group">
            <label>New Password : </label>
            <input
              type="password"
              class="form-control"
              formControlName="password"
            />
            <div>
              <span
                class="alert alert-danger"
                *ngIf="
                  this.SignUpForm.get('password').invalid &&
                  this.SignUpForm.get('password').touched
                "
                >Password should be minimum 8 and maximum 16 characters
                long</span
              >
            </div>
          </div>

          <div class="form-group">
            <label>User Role : </label>
            <select
              class="form-control"
              formControlName="userRole"
              ngDefaultControl=""
            >
              <option value="">Select</option>
              <option value="User">User</option>
              <option value="Admin" disabled>Admin</option>
            </select>
            <div>
              <span
                class="alert alert-danger"
                *ngIf="
                  this.SignUpForm.get('userRole').invalid &&
                  this.SignUpForm.get('userRole').touched
                "
                >Please select user role</span
              >
            </div>
          </div>
```

```
          <br />
          <div class="form-group text-center">
            <button class="btn btn-primary" [routerLink]="['/']">Login</button
            >&nbsp;
            <button class="btn btn-primary" [disabled]="SignUpForm.invalid">
              Create User
            </button>
          </div>
        </div>
      </form>
    </div>
  </div>
</div>
```

sign-up.component.ts

```
import { Component, OnInit } from '@angular/core';
import { FormControl, FormGroup, Validators } from '@angular/forms';
import { Router } from '@angular/router';
import { ErrorEnum } from 'src/app/model/error-enum';
import { UserModel } from 'src/app/model/user-model';
import { UserService } from 'src/app/services/user.service';

@Component({
  selector: 'app-sign-up',
  templateUrl: './sign-up.component.html',
  styleUrls: ['./sign-up.component.css'],
})
export class SignUpComponent implements OnInit {
  SignUpForm!: FormGroup;
  userResp: any[];

  constructor(private userService: UserService, private router: Router) {}

  ngOnInit(): void {
    sessionStorage.removeItem('userId');
    sessionStorage.removeItem('password');
    sessionStorage.removeItem('userRole');

    this.SignUpForm = new FormGroup({
      userId: new FormControl('', Validators.required),
      password: new FormControl('', [
        Validators.required,
        Validators.minLength(5),
        Validators.maxLength(16),
      ]),
      userRole: new FormControl('', Validators.required),
    });
  }

  onSubmit() {
    let userModel = new UserModel(
      this.SignUpForm.value.userId,
      this.SignUpForm.value.password
    );

    this.userService.signingInUser(userModel).subscribe(
      (res) => {
        this.userResp = res;
        if (this.userResp.length == 0) {
          userModel.userRole = this.SignUpForm.value.userRole;

          this.userService.registerUser(userModel).subscribe(
            (res) => {
              // alert(ErrorEnum.REGISTRATION_SUCCESS);
              this.router.navigate(['/']);
            },
            (error) => {
              alert(ErrorEnum.JSON_CONNECTION_FAILED);
            }
          );
        } else {
          this.SignUpForm.reset();
          alert(ErrorEnum.USER_ALREADY_EXISTS);
          return;
        }
      },
      (error) => {
        alert(ErrorEnum.JSON_CONNECTION_FAILED);
      }
    );
  }
}
```