//src\app\admin-agent-packages\admin-agent-packages.component.css

/\* Center the heading and add spacing \*/

h2 {

    text-align: center;

    margin: 20px 0;

    color: #333;

    font-size: 24px;

  }

  /\* Style the table \*/

  table {

    width: 90%;

    margin: 0 auto;

    border-collapse: collapse;

  }

  th, td {

    padding: 12px;

    text-align: left;

    border-bottom: 1px solid #ddd;

  }

  th {

    background-color: #f8f8f8;

    color: #444;

  }

  tr:hover {

    background-color: #f1f1f1;

  }

  /\* Message if no packages found \*/

  .no-packages {

    text-align: center;

    color: gray;

    margin-top: 30px;

    font-size: 18px;

  }

//src\app\admin-agent-packages\admin-agent-packages.component.html

<div class="container">

  <h2>Packages Listed by Agent #{{ agentId }}</h2>

<!-- can add more data here to show in the frontend -->

  <table \*ngIf="packages.length > 0; else noPackages">

    <thead>

      <tr>

        <th>Title</th>

        <th>Description</th>

        <th>Price (₹)</th>

        <th>Duration (days)</th>

        <th>Start Date</th>

        <th>End Date</th>

        <th>Max Capacity</th>

      </tr>

    </thead>

    <tbody>

      <tr \*ngFor="let pkg of packages">

        <td>{{ pkg.title }}</td>

        <td>{{ pkg.description }}</td>

        <td>{{ pkg.price }}</td>

        <td>{{ pkg.duration }}</td>

        <td>{{ pkg.tripStartDate }}</td>

        <td>{{ pkg.tripEndDate }}</td>

        <td>{{ pkg.maxCapacity }}</td>

      </tr>

    </tbody>

  </table>

  <ng-template #noPackages>

    <p>No packages found for this agent.</p>

  </ng-template>

</div>

//src\app\admin-agent-packages\admin-agent-packages.component.spec.ts

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { AdminAgentPackagesComponent } from './admin-agent-packages.component';

describe('AdminAgentPackagesComponent', () => {

  let component: AdminAgentPackagesComponent;

  let fixture: ComponentFixture<AdminAgentPackagesComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [AdminAgentPackagesComponent]

    })

    .compileComponents();

    fixture = TestBed.createComponent(AdminAgentPackagesComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

//src\app\admin-agent-packages\admin-agent-packages.component.ts

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

import { CommonModule } from '@angular/common';

import { HttpClient, HttpHeaders } from '@angular/common/http';

import { ActivatedRoute } from '@angular/router';

@Component({

  selector: 'app-admin-agent-packages',

  standalone: true,

  imports: [CommonModule],

  templateUrl: './admin-agent-packages.component.html',

  styleUrls: ['./admin-agent-packages.component.css']

})

export class AdminAgentPackagesComponent implements OnInit {

  agentId!: number; // ! tells ts we will initiate it later

  packages: any[] = [];

  constructor(

    private route: ActivatedRoute,

    private http: HttpClient

  ) {}

  ngOnInit(): void {

    // Get agent ID from route parameter

    this.agentId = Number(this.route.snapshot.paramMap.get('id'));

    console.log("Agent ID:",this.agentId);

    // Call backend to fetch agent's packages

    const token = localStorage.getItem('token');

    const headers = new HttpHeaders().set('Authorization', `Bearer ${token}`);

    this.http

    .get<any>(`http://localhost:8080/api/packages/agent/${this.agentId}`)

      .subscribe({

        next: (res) => {

          console.log("Fetched packages", res);

          this.packages = res.data; // res is the full object, data contains array

        },

        error: (err) => {

          console.error('Failed to fetch packages', err);

          this.packages = [];

        }

      });

  }

}

//src\app\admin-dashboard\admin-dashboard.component.css

.dashboard {

    padding: 20px;

    text-align: center;

  }

  .summary {

    margin: 20px auto;

    font-size: 18px;

    line-height: 1.6;

  }

  .buttons {

    display: flex;  /\*Arrange buttons in a row\*/

    justify-content: center; /\*Center the buttons horizontally\*/

    gap: 16px; /\* Add spaCE BETWEEN the buttons\*/

    margin-top: 20px;

  }

  .buttons button {

    margin-top: 0 10px;

    padding: 10px 20px;

    font-weight: bold;

    background-color: #4caf50;

    color: white;

    border: none;

    border-radius: 5px;

    cursor: pointer;

  }

//src\app\admin-dashboard\admin-dashboard.component.html

<div class="dashboard">

  <h2>Admin Dashboard</h2>

  <p>Welcome, Admin! Here's a summary of platform users:</p>

  <div class="summary">

    <p><strong>Total Users (excluding admin):</strong> {{ totalUsers }}</p>

    <p><strong>Total Agents:</strong> {{ totalAgents }}</p>

    <p><strong>Total Customers:</strong> {{ totalCustomers }}</p>

  </div>

  <div class="buttons">

    <button (click)="goToProfile()">Go to My Profile</button>

    <button (click)="goToManageUsers()">Manage Users</button>

  </div>

</div>

//src\app\admin-dashboard\admin-dashboard.component.spec.ts

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { AdminDashboardComponent } from './admin-dashboard.component';

describe('AdminDashboardComponent', () => {

  let component: AdminDashboardComponent;

  let fixture: ComponentFixture<AdminDashboardComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [AdminDashboardComponent]

    })

    .compileComponents();

    fixture = TestBed.createComponent(AdminDashboardComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

//src\app\admin-dashboard\admin-dashboard.component.ts

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

import { HttpClient } from '@angular/common/http';

import { ToastrService } from 'ngx-toastr';

import { Router } from '@angular/router';

interface UserRoleCountResponse {

  totalUsers: number;

  agentCount: number;

  customerCount: number;

}

@Component({

  selector: 'app-admin-dashboard',

  templateUrl: './admin-dashboard.component.html',

  styleUrls: ['./admin-dashboard.component.css']

})

export class AdminDashboardComponent implements OnInit {

  totalUsers: number = 0;

  totalAgents: number = 0;

  totalCustomers: number = 0;

  constructor(

    private http: HttpClient,

    private toastr: ToastrService,

    private router: Router

  ) {}

  ngOnInit(): void {

    this.fetchUserCounts();

  }

  fetchUserCounts() {

    const token = localStorage.getItem('token');

this.http.get<UserRoleCountResponse>('http://localhost:8080/api/users/counts', {

      headers: { Authorization: `Bearer ${token}` }

    }).subscribe({

      next: (res) => {

        this.totalUsers = res.totalUsers;

        this.totalAgents = res.agentCount;

        this.totalCustomers = res.customerCount;

      },

      error: () => this.toastr.error('Failed to fetch user counts.')

    });

  }

  goToProfile() {

    this.router.navigate(['/my-profile']);

  }

  goToManageUsers() {

    this.router.navigate(['/admin-user-manage']);

  }

}

//src\app\admin-user-manage\admin-user-manage.component.css

/\* Container spacing \*/

.container {

  padding: 20px;

  font-family: Arial, sans-serif;

}

/\* Table layout \*/

table {

  width: 100%;

  border-collapse: collapse;

  margin-top: 20px;

}

/\* Table headers \*/

th {

  background-color: #f0f0f0;

  text-align: left;

  padding: 10px;

  font-weight: 600;

  border-bottom: 1px solid #ddd;

}

/\* Table rows \*/

td {

  padding: 10px;

  border-bottom: 1px solid #ddd;

  vertical-align: middle;

}

/\* Buttons in actions/view column \*/

button {

  padding: 5px 10px;

  margin-right: 5px;

  background-color: #007bff;

  color: white;

  border: none;

  border-radius: 4px;

  cursor: pointer;

}

button:hover {

  background-color: #0056b3;

}

/\* Align buttons in same row \*/

td button + button {

  margin-left: 5px;

}

/\* Make select box look nicer \*/

select {

  padding: 5px;

  border-radius: 4px;

  border: 1px solid #ccc;

}

h2 {

  text-align: center;

  margin-top: 20px;

  margin-bottom: 20px;

  font-size: 24px;

  color: #333;

}

//src\app\admin-user-manage\admin-user-manage.component.html

<div class="manage-container">

    <h2>All Registered Users</h2>

    <table border="1">

      <thead>

        <tr>

          <th>Name</th>

          <th>Email</th>

          <th>Phone</th>

          <th>View</th>

          <th>Role</th>

          <th>Actions</th>

        </tr>

        </thead>

        <tbody>

        <tr \*ngFor="let user of users">

          <td>{{ user.name }}</td>

          <td>{{ user.email }}</td>

          <td>{{ user.contactNumber }}</td>

          <td>

            <a \*ngIf="user.role === 'AGENT'"

            [routerLink]="['/admin/agent-packages', user.id]">

            <button>View Packages</button>

            </a>

          <td>

            <select [value]="user.role"

            (change)="onRoleChange(user, $event)">

              <option value="CUSTOMER">CUSTOMER</option>

              <option value="AGENT">AGENT</option>

          </select>

          </td>

          <td>

            <button (click)="deleteUser(user.id)">Delete</button>

          </td>

        </tr>

      </tbody>

    </table>

  </div>

//src\app\admin-user-manage\admin-user-manage.component.spec.ts

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { AdminUserManageComponent } from './admin-user-manage.component';

describe('AdminUserManageComponent', () => {

  let component: AdminUserManageComponent;

  let fixture: ComponentFixture<AdminUserManageComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [AdminUserManageComponent]

    })

    .compileComponents();

    fixture = TestBed.createComponent(AdminUserManageComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

//src\app\admin-user-manage\admin-user-manage.component.ts

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

import { HttpClient, HttpHeaders, HttpClientModule } from '@angular/common/http';

import { ToastrService } from 'ngx-toastr';

import { CommonModule } from '@angular/common';

import { Router } from '@angular/router';

import { RouterModule } from '@angular/router';

@Component({

  selector: 'app-admin-user-manage',

  standalone: true,

  imports: [CommonModule, RouterModule, HttpClientModule],

  templateUrl: './admin-user-manage.component.html',

  styleUrls: ['./admin-user-manage.component.css']

})

export class AdminUserManageComponent implements OnInit {

  users: any[] = [];

  constructor(private router: Router, private http: HttpClient, private toastr: ToastrService) {}

  ngOnInit(): void {

    this.fetchUsers();

  }

  // 🔹 Fetch all users from backend

  fetchUsers(): void {

    const token = localStorage.getItem('token');

    this.http.get<any[]>('http://localhost:8080/api/users', {

      headers: { Authorization: `Bearer ${token}` }

    }).subscribe({

      //next: (res) => this.users = res,

      next: (res) => {

        this.users = res.filter(user => user.role && user.role.toUpperCase() !== 'ADMIN');

      },

      error: () => this.toastr.error('Failed to load users')

    });

  }

  // Called when admin changes the role of a user from dropdown

onRoleChange(user: any, event:Event): void {

  const selectedRole = (event.target as HTMLSelectElement).value;

  // Prevent manually assigning ADMIN role

  if (selectedRole === 'ADMIN') {

    this.toastr.warning('Admin role cannot be assigned manually.');

    return;

  }

  // Get token from localStorage

  const token = localStorage.getItem('token');

  // Create a copy of user with updated role

  const updatedUser = { ...user, role: selectedRole };

  // Make PUT request to update role

this.http.put(`http://localhost:8080/api/users/${user.id}`, updatedUser, {

    headers: {

      Authorization: `Bearer ${token}`

    }

  }).subscribe({

    next: () => {

      this.toastr.success('Role updated successfully');

      this.fetchUsers(); // Refresh the user list after update

    },

    error: () => {

      this.toastr.error('Failed to update role');

    }

  });

}

  // 🔹 Delete user

  deleteUser(id: number): void {

    const token = localStorage.getItem('token');

  this.http.delete(`http://localhost:8080/api/users/${id}`, {

      headers: { Authorization: `Bearer ${token}` },

      responseType: 'text', //avoids json parsing

      observe: 'response'  //  Observe the full HTTP response

    }).subscribe({

      next: (response) => {

        // Check if status is 200 OK or 204 No Content

        if (response.status === 200 || response.status === 204) {

          this.toastr.success('User deleted successfully');

          window.location.reload(); // ⟳ Refresh the page to update the UI

        } else {

          this.toastr.error('Unexpected response from server');

        }

      },

      error: (err) => {

        console.error("❌ Delete failed:", err);

        this.toastr.error('Failed to delete user');

      }

    });

  }

  //view agent packages

  viewAgentPackages(agentId: number) {

    this.router.navigate(['/admin/agent-packages',agentId]);

  }

  /\* deleteUser(id: number): void {

    const token = localStorage.getItem('token');

this.http.delete(`http://localhost:8080/api/users/${id}`, {

      headers: { Authorization: `Bearer ${token}` }

    }).subscribe({

      next: () => {

        //removes user from local list

        //this.users = this.users.filter(user => user.id !== id);

        this.toastr.success('User deleted');

        window.location.reload();

      },

      error: () => this.toastr.error('Failed to delete user')

    });

  }\*/

}

//src\app\agent-dashboard\agent-dashboard.component.css

.dashboard-container {

    padding: 20px;

    font-family: Arial, sans-serif;

  }

  button {

    margin: 5px;

    padding: 8px 14px;

    border: none;

    background-color: #2196f3;

    color: white;

    border-radius: 4px;

    cursor: pointer;

  }

  button:hover {

    background-color: #0b7dda;

  }

  ul {

    list-style: none;

    padding: 0;

  }

  li {

    margin: 10px 0;

    padding: 10px;

    background: #f1f1f1;

    border-radius: 5px;

  }

//src\app\agent-dashboard\agent-dashboard.component.html

<div class="dashboard-container">

    <h2>Welcome Agent!</h2>

    <button (click)="showCreateForm()">➕ Create New Package</button>

    <button (click)="goToProfile()">👤 My Profile</button>

    <hr>

    <div \*ngIf="packages.length > 0; else noPackages">

      <h3>Your Travel Packages:</h3>

      <ul>

        <li \*ngFor="let pkg of packages">

          <strong>{{ pkg.title }}</strong> – {{ pkg.destination }} (₹{{ pkg.price }})

          <!--view details of the package-->

          <button [routerLink]="['/view-package', pkg.packageId]">View Details</button>

          <button [routerLink]="['/edit-package', pkg.packageId]">Edit</button>

          <button (click)="deletePackage(pkg.packageId)">🗑️ Delete</button>

        </li>

      </ul>

    </div>

    <ng-template #noPackages>

      <p>No packages found. Click "Create New Package" to add one!</p>

    </ng-template>

    <hr/>

  <!-- CREATE NEW PACKAGE FORM -->

  <div \*ngIf="formVisible">

    <h3>Create New Travel Package</h3>

    <form (ngSubmit)="submitPackage()" #pkgForm="ngForm">

      <!-- Basic Info -->

      <label>Title: </label>

      <input type="text" placeholder="Title" [(ngModel)]="newPackage.title" name="title" required><br>

      <label>Description: </label><br>

      <textarea placeholder="Description" [(ngModel)]="newPackage.description" name="description" required></textarea><br>

      <label>Duration: </label>

      <input type="number" placeholder="Duration (days)" [(ngModel)]="newPackage.duration" name="duration"><br>

      <label>Price: </label>

      <input type="number" placeholder="Price" [(ngModel)]="newPackage.price" name="price"><br>

      <label>Max Capacity: </label>

      <input type="number" placeholder="Max Capacity" [(ngModel)]="newPackage.maxCapacity" name="maxCapacity"><br>

      <label>Country: </label>

      <input type="text" placeholder="Country" [(ngModel)]="newPackage.country" name="country"><br>

      <label>Destination: </label>

      <input type="text" placeholder="Destination" [(ngModel)]="newPackage.destination" name="destination"><br>

      <label>Trip Type: </label>

       <select [(ngModel)]="newPackage.tripType" name="tripType" required>

        <option value="">-- Select Trip Type --</option>

        <option value="International Trip">International Trip</option>

        <option value="Domestic Trip">Domestic Trip</option>

       </select><br>

      <label>Trip Start Date:</label>

      <input type="date" [(ngModel)]="newPackage.tripStartDate" name="tripStartDate"><br>

      <label>Trip End Date:</label>

      <input type="date" [(ngModel)]="newPackage.tripEndDate" name="tripEndDate"><br>

      <!--Offer Section-->

      <h4>Offer Details</h4>

      <label>Offer Description: </label>

      <textarea placeholder="Offer Description" [(ngModel)]="newPackage.offer.description" name="offerDesc"></textarea><br>

      <label>Coupon Code: </label>

      <input type="text" placeholder="Coupon Code" [(ngModel)]="newPackage.offer.couponCode" name="Coupon Code"><br>

      <label>Discount Percentage: </label>

      <input type="number" placeholder="Discount (%)" [(ngModel)]="newPackage.offer.discountPercentage" name="discountPercentage"><br>

      <label><input type="checkbox" [(ngModel)]="newPackage.offer.active" name="offerActive">Offer Active</label><br>

      <!--Flights-->

      <h4>Flight Details</h4>

      <div \*ngFor="let flight of newPackage.flights; let i=index">

        <label>Airline: </label>

        <input type="text" placeholder="Airline" [(ngModel)]="flight.airline" [name]="'airline'+i"><br>

        <label>From City: </label>

        <input type="text" placeholder="From City" [(ngModel)]="flight.fromCity" [name]="'fromCity'+i"><br>

        <label>To City: </label>

        <input type="text" placeholder="To City" [(ngModel)]="flight.toCity" [name]="'toCity'+i"><br>

        <label>Departire Time: </label>

        <input type="text" placeholder="Departure Time" [(ngModel)]="flight.departureTime" [name]="'departureTime'+i"><br>

        <label>Arrival Time: </label>

        <input type="text" placeholder="Arrival Time" [(ngModel)]="flight.arrivalTime" [name]="'arrivalTime'+i"><br>

      </div>

      <!-- Hotels -->

      <h4>Hotel Details</h4>

      <div \*ngFor="let hotel of newPackage.hotels; let i=index">

        <label>Hotel Name: </label>

        <input type="text" placeholder="Hotel Name" [(ngModel)]="hotel.name" [name]="'hotelName'+i"><br>

        <label>City: </label>

        <input type="text" placeholder="City" [(ngModel)]="hotel.city" [name]="'hotelCity'+i"><br>

        <label>Rating: </label>

        <input type="number" placeholder="Rating" [(ngModel)]="hotel.rating" [name]="'hotelRating'+i"><br>

        <label>Nights Staying: </label>

        <input type="number" placeholder="Nights" [(ngModel)]="hotel.nights" [name]="'hotelNights'+i"><br>

        <label>Cost per Night: </label>

        <input type="number" placeholder="Cost/Night" [(ngModel)]="hotel.costPerNight" [name]="'hotelCost'+i"><br>

      </div>

      <!-- Sightseeing -->

      <h4>Sightseeing</h4>

      <div \*ngFor="let sight of newPackage.sightseeing; let i=index">

        <label>Location: </label>

        <input type="text" placeholder="Location" [(ngModel)]="sight.location" [name]="'sightLoc'+i"><br>

        <label>Sight Description: </label>

        <input type="text" placeholder="Description" [(ngModel)]="sight.description" [name]="'sightDesc'+i"><br>

        <button type="button" (click)="removeSightseeing(i)" \*ngIf="newPackage.sightseeing.length > 1">

          "Remove Sightseeing"

        </button>

      </div>

      <button type="button" (click)="addSightseeing()">+ Add Sightseeing</button>

      <!-- Itinerary -->

      <h4>Itinerary</h4>

      <div \*ngFor="let item of newPackage.itinerary; let i=index">

        <label>Day Number: </label>

        <input type="number" placeholder="Day #" [(ngModel)]="item.dayNumber" [name]="'day'+i" placeholder="Day #" required><br>

        <label>Activity Title: </label>

        <input type="text" placeholder="Activity Title" [(ngModel)]="item.activityTitle" [name]="'activityTitle'+i" placeholder="Activity Description" required><br>

        <label>Activity Description: </label>

        <input type="text" placeholder="Activity Description" [(ngModel)]="item.activityDescription" [name]="'activityDesc'+i" required><br>

        <button type="button" (click)="removeItineraryDay(i)" \*ngIf="newPackage.itinerary.length>1">Remove Day</button>

        <hr>

      </div>

      <button type="button" (click)="addItineraryDay()">+Add Itinerary Day</button>

      <!-- Submit and Cancel -->

      <br />

      <button type="submit">Submit Package</button>

      <button type="button" (click)="cancelForm()">Cancel</button>

    </form>

  </div>

</div>

//src\app\agent-dashboard\agent-dashboard.component.spec.ts

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { AgentDashboardComponent } from './agent-dashboard.component';

describe('AgentDashboardComponent', () => {

  let component: AgentDashboardComponent;

  let fixture: ComponentFixture<AgentDashboardComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [AgentDashboardComponent]

    })

    .compileComponents();

    fixture = TestBed.createComponent(AgentDashboardComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

//src\app\agent-dashboard\agent-dashboard.component.ts

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

import { HttpClient, HttpClientModule } from '@angular/common/http';

import { CommonModule } from '@angular/common';

import { FormsModule } from '@angular/forms';

import { RouterLink } from '@angular/router';

@Component({

  selector: 'app-agent-dashboard',

  standalone: true,

  imports: [CommonModule, HttpClientModule, FormsModule, RouterLink],

  templateUrl: './agent-dashboard.component.html',

  styleUrl: './agent-dashboard.component.css'

})

export class AgentDashboardComponent implements OnInit{

  packages: any[] = []; //store the list of travel packages

  formVisible: boolean = false;

  agentId :number=0;

  newPackage: any = {

    title:'',

    description:'',

    duration:'',

    price:'',

    maxCapacity:'',

    country:'',

    destination:'',

    tripType:'',

    tripStartDate:'',

    tripEndDate:'',

    active: true,

    offer: { couponCode:'', description:'', discountPercentage: 0, active: false},

    flights: [

      {airline: '', fromCity:'', toCity:'', departureTime:'', arrivalTime:''}

    ],

    hotels: [

      {name:'', city:'', rating:'', nights:'', costPerNight:''}

    ],

    sightseeing: [

      {location:'', description:''}

    ],

    itinerary: [

      {dayNumber:'', activityTitle:'', activityDescription:''}

    ]

  };

  private http = inject(HttpClient);

  ngOnInit(): void {

    const storedId = localStorage.getItem('userId');

    this.agentId = storedId ? parseInt(storedId, 10) : 0;

    if (this.agentId>0) {

      this.loadPackages();

    } else {

      console.error('Agent ID not found in localStorage');

      // optionally redirect to login page or show error

    }

  }

  //Get packages created by this agent

  loadPackages(){

    const url = `http://localhost:8080/api/packages/agent/${this.agentId}`;

    this.http.get<any>(url).subscribe({

      next: (res) =>{

        this.packages = res.data;

        console.log("Loaded packages:", this.packages);//optional debug

      } ,

      error: (err) => console.error('Error loading packages', err)

    });

  }

  //Submit new package to backend

  submitPackage(){

    const url= `http://localhost:8080/api/packages`;

    const fullData = {

      ...this.newPackage,

      agentId: this.agentId

    };

    this.http.post(url, fullData).subscribe({

      next: () => {

        alert('Package created successfully!');

        this.formVisible = false;

        this.loadPackages();

        this.resetForm();

      },

      error: (err) => {

        console.error('Error correcting package', err);

        alert('Failed to create package. Check your input.');

      }

    });

  }

  //Delete a package

  deletePackage(id: number){

    if(!id){

      console.error('Invalid package ID:',id);

      alert('Error: Cannot delete package. Invalid ID.');

      return ;

    }

    const confirmDelete = confirm("Are you sure you want to delete this package?");

    if(!confirmDelete) return;

    const url = `http://localhost:8080/api/packages/${id}`;

    this.http.delete(url).subscribe({

      next: () => {

        alert('Package deleted successfully!');

        this.loadPackages();

      },

      error: (err) => {

        console.error('Error deleting package', err);

        alert('Failed to delete Package.')

      }

    });

  }

  //show or hide create form

  showCreateForm(){

    this.formVisible = true;

  }

  cancelForm() {

    this.formVisible = false;

  }

  //placeholder for edit

  editPackage(pkg: any){

    alert('Edit package: ${pkg.title} - feature coming soon!');

  }

  //Reset form after success

  resetForm() {

    this.newPackage = {

      title:'',

      description:'',

      duration:'',

      price:'',

      maxCapacity:'',

      country:'',

      destination:'',

      tripType:'',

      tripStartDate:'',

      tripEndDate:'',

      active: true,

      offer: { couponCode:'', description:'', discountPercentage: 0, active: false},

      flights: [

        {airline: '', fromCity:'', toCity:'', departureTime:'', arrivalTime:''}

      ],

      hotels: [

        {name:'', city:'', rating:'', nights:'', costPerNight:''}

      ],

      sightseeing: [

        {location:'', description:''}

      ],

      itinerary: [

        {dayNumber:'', activityTitle:'', activityDescription:''}

      ]

    };

  }

  addItineraryDay(){

    const newDay = this.newPackage.itinerary.length+1;

    this.newPackage.itinerary.push({

      dayNumber: newDay,

      activityTitle: '',

      activityDescription: ''

    });

  }

  removeItineraryDay(index:number): void{

    this.newPackage.itinerary.splice(index,1);

    this.newPackage.itinerary.forEach((item: any, i: number) =>{

      item.dayNumber = i+1;

    });

  }

  // for sight seeing

  addSightseeing(): void {

    this.newPackage.sightseeing.push({location:'', description:''});

  }

  removeSightseeing(index: number): void {

    this.newPackage.sightseeing.splice(index,1);

  }

  //Go to my profile

  goToProfile(){

    window.location.href = '/my-profile';

  }

}

//src\app\customer-dashboard\customer-dashboard.component.css

//src\app\customer-dashboard\customer-dashboard.component.html

<p>customer-dashboard works!</p>

<a routerLink="/my-profile">My Profile</a>

//src\app\customer-dashboard\customer-dashboard.component.spec.ts

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { CustomerDashboardComponent } from './customer-dashboard.component';

describe('CustomerDashboardComponent', () => {

  let component: CustomerDashboardComponent;

  let fixture: ComponentFixture<CustomerDashboardComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [CustomerDashboardComponent]

    })

    .compileComponents();

    fixture = TestBed.createComponent(CustomerDashboardComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

//src\app\customer-dashboard\customer-dashboard.component.ts

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

@Component({

  selector: 'app-customer-dashboard',

  imports: [],

  templateUrl: './customer-dashboard.component.html',

  styleUrl: './customer-dashboard.component.css'

})

export class CustomerDashboardComponent {

}

//src\app\edit-package\edit-package.component.css

//src\app\edit-package\edit-package.component.html

<div class="edit-container">

    <h2>Edit Travel Package</h2>

    <form (ngSubmit)="submitPackage()" #pkgForm="ngForm">

        <!-- Basic Info -->

        <label>Title: </label>

        <input type="text" placeholder="Title" [(ngModel)]="newPackage.title" name="title" required><br>

        <label>Description: </label><br>

        <textarea placeholder="Description" [(ngModel)]="newPackage.description" name="description" required></textarea><br>

        <label>Duration: </label>

        <input type="number" placeholder="Duration (days)" [(ngModel)]="newPackage.duration" name="duration"><br>

        <label>Price: </label>

        <input type="number" placeholder="Price" [(ngModel)]="newPackage.price" name="price"><br>

        <label>Max Capacity: </label>

        <input type="number" placeholder="Max Capacity" [(ngModel)]="newPackage.maxCapacity" name="maxCapacity"><br>

        <label>Country: </label>

        <input type="text" placeholder="Country" [(ngModel)]="newPackage.country" name="country"><br>

        <label>Destination: </label>

        <input type="text" placeholder="Destination" [(ngModel)]="newPackage.destination" name="destination"><br>

        <label>Trip Type: </label>

         <select [(ngModel)]="newPackage.tripType" name="tripType" required>

          <option value="">-- Select Trip Type --</option>

          <option value="International Trip">International Trip</option>

          <option value="Domestic Trip">Domestic Trip</option>

         </select><br>

        <label>Trip Start Date:</label>

        <input type="date" [(ngModel)]="newPackage.tripStartDate" name="tripStartDate"><br>

        <label>Trip End Date:</label>

        <input type="date" [(ngModel)]="newPackage.tripEndDate" name="tripEndDate"><br>

        <!--Offer Section-->

        <h4>Offer Details</h4>

        <label>Offer Description: </label>

        <textarea placeholder="Offer Description" [(ngModel)]="newPackage.offer.description" name="offerDesc"></textarea><br>

        <label>Coupon Code: </label>

        <input type="text" placeholder="Coupon Code" [(ngModel)]="newPackage.offer.couponCode" name="Coupon Code"><br>

        <label>Discount Percentage: </label>

        <input type="number" placeholder="Discount (%)" [(ngModel)]="newPackage.offer.discountPercentage" name="discountPercentage"><br>

        <label><input type="checkbox" [(ngModel)]="newPackage.offer.active" name="offerActive">Offer Active</label><br>

        <!--Flights-->

        <h4>Flight Details</h4>

        <div \*ngFor="let flight of newPackage.flights; let i=index">

          <label>Airline: </label>

          <input type="text" placeholder="Airline" [(ngModel)]="flight.airline" [name]="'airline'+i"><br>

          <label>From City: </label>

          <input type="text" placeholder="From City" [(ngModel)]="flight.fromCity" [name]="'fromCity'+i"><br>

          <label>To City: </label>

          <input type="text" placeholder="To City" [(ngModel)]="flight.toCity" [name]="'toCity'+i"><br>

          <label>Departire Time: </label>

          <input type="text" placeholder="Departure Time" [(ngModel)]="flight.departureTime" [name]="'departureTime'+i"><br>

          <label>Arrival Time: </label>

          <input type="text" placeholder="Arrival Time" [(ngModel)]="flight.arrivalTime" [name]="'arrivalTime'+i"><br>

        </div>

        <!-- Hotels -->

        <h4>Hotel Details</h4>

        <div \*ngFor="let hotel of newPackage.hotels; let i=index">

          <label>Hotel Name: </label>

          <input type="text" placeholder="Hotel Name" [(ngModel)]="hotel.name" [name]="'hotelName'+i"><br>

          <label>City: </label>

          <input type="text" placeholder="City" [(ngModel)]="hotel.city" [name]="'hotelCity'+i"><br>

          <label>Rating: </label>

          <input type="number" placeholder="Rating" [(ngModel)]="hotel.rating" [name]="'hotelRating'+i"><br>

          <label>Nights Staying: </label>

          <input type="number" placeholder="Nights" [(ngModel)]="hotel.nights" [name]="'hotelNights'+i"><br>

          <label>Cost per Night: </label>

          <input type="number" placeholder="Cost/Night" [(ngModel)]="hotel.costPerNight" [name]="'hotelCost'+i"><br>

        </div>

        <!-- Sightseeing -->

        <h4>Sightseeing</h4>

        <div \*ngFor="let sight of newPackage.sightseeing; let i=index">

          <label>Location: </label>

          <input type="text" placeholder="Location" [(ngModel)]="sight.location" [name]="'sightLoc' +i"><br>

          <label>Sight Description: </label>

          <input type="text" placeholder="Description" [(ngModel)]="sight.description" [name]="'sightDesc' +i"><br>

          <button type="button" (click)="removeSightseeing(i)" \*ngIf="newPackage.sightseeing.length >1 ">

            Remove Sightseeing

          </button>

          <hr>

        </div>

        <button type="button" (click)="addSightseeing()">+ Add Sightseeing</button>

        <!-- Itinerary -->

        <h4>Itinerary</h4>

        <div \*ngFor="let item of newPackage.itinerary; let i=index">

          <label>Day Number: </label>

          <input type="number" placeholder="Day #" [(ngModel)]="item.dayNumber" [name]="'day'+i" placeholder="Day #" required><br>

          <label>Activity Title: </label>

          <input type="text" placeholder="Activity Title" [(ngModel)]="item.activityTitle" [name]="'activityTitle'+i" placeholder="Activity Description" required><br>

          <label>Activity Description: </label>

          <input type="text" placeholder="Activity Description" [(ngModel)]="item.activityDescription" [name]="'activityDesc'+i" required><br>

          <button type="button" (click)="removeItineraryDay(i)" \*ngIf="newPackage.itinerary.length>1">Remove Day</button>

          <hr>

        </div>

        <button type="button" (click)="addItineraryDay()">+Add Itinerary Day</button>

        <!-- Submit and Cancel -->

        <br />

        <button type="submit">Submit Package</button>

        <button type="button" (click)="cancelForm()">Cancel</button>

      </form>

  </div>

//src\app\edit-package\edit-package.component.spec.ts

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { EditPackageComponent } from './edit-package.component';

describe('EditPackageComponent', () => {

  let component: EditPackageComponent;

  let fixture: ComponentFixture<EditPackageComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [EditPackageComponent]

    })

    .compileComponents();

    fixture = TestBed.createComponent(EditPackageComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

//src\app\edit-package\edit-package.component.ts

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

import { ActivatedRoute, Router } from '@angular/router';

import { HttpClient } from '@angular/common/http';

import { CommonModule } from '@angular/common';

import { FormsModule } from '@angular/forms';

import { HttpClientModule } from '@angular/common/http';

import { RouterModule } from '@angular/router';

@Component({

  selector: 'app-edit-package',

  standalone: true,

  imports: [CommonModule, FormsModule, HttpClientModule, RouterModule],

  templateUrl: './edit-package.component.html',

  styleUrls: ['./edit-package.component.css']

})

export class EditPackageComponent implements OnInit {

  packageId!: number;

  // packageData: any = {};

  newPackage: any = {

    title:'',

    description:'',

    duration:'',

    price:'',

    maxCapacity:'',

    country:'',

    destination:'',

    tripType:'',

    tripStartDate:'',

    tripEndDate:'',

    active: true,

    offer: { couponCode:'', description:'', discountPercentage: 0, active: false},

    flights: [

      {airline: '', fromCity:'', toCity:'', departureTime:'', arrivalTime:''}

    ],

    hotels: [

      {name:'', city:'', rating:'', nights:'', costPerNight:''}

    ],

    sightseeing: [

      {location:'', description:''}

    ],

    itinerary: [

      {dayNumber:'', activityTitle:'', activityDescription:''}

    ]

  };

  constructor(

    private route: ActivatedRoute,

    private http: HttpClient,

    private router: Router

  ) {}

  ngOnInit(): void {

    // Capture ID from URL

    this.packageId = Number(this.route.snapshot.paramMap.get('id'));

    // Fetch existing package data

    const url=`http://localhost:8080/api/packages/${this.packageId}`;

      this.http.get<any>(url).subscribe({

        next: (res) => {

          this.newPackage = res.data;

        },

        error: (err) => {

          console.error('Failed to fetch package',err);

          alert('Error loading package data.');

        }

      });

  }

  submitPackage(){

    const url=`http://localhost:8080/api/packages/${this.packageId}`;

      this.http.put(url, this.newPackage).subscribe({

        next: () => {

          alert('Package updated successfully!');

          this.router.navigate(['/agent-dashboard']);

        },

        error: (err) =>

        { console.error('Failed to update package',err);

          alert('Error updating package');

        }

      });

  }

  addItineraryDay(){

    const nextDay = this.newPackage.itinerary.length+1;

    this.newPackage.itinerary.push({

      dayNumber: nextDay,

      activityTitle: '',

      activityDescription: ''

    });

  }

  removeItineraryDay(index:number): void{

    this.newPackage.itinerary.splice(index,1);

    this.newPackage.itinerary.forEach((item: any, i: number) =>{

      item.dayNumber = i+1;

    });

  }

  // for sight seeing

  addSightseeing(): void {

    this.newPackage.sightseeing.push({location:'', description:''});

  }

  removeSightseeing(index: number): void {

    this.newPackage.sightseeing.splice(index,1);

  }

  cancelForm(){

    this.router.navigate(['/agent-dashboard']);

  }

}

//src\app\forgot-password\forgot-password.component.css

.forgot-password-container {

    width: 300px;

    margin: 100px auto;

    padding: 20px;

    background-color: #f8f9fa;

    text-align: center;

    border-radius: 8px;

    box-shadow: 0px 0px 10px #ccc;

  }

  input {

    width: 100%;

    padding: 8px;

    margin-top: 8px;

    margin-bottom: 12px;

    border: 1px solid #ccc;

    border-radius: 4px;

  }

  button {

    padding: 10px 20px;

    background-color: #1f8ef1;

    color: white;

    border: none;

    border-radius: 4px;

    cursor: pointer;

  }

  .forgot-password-link {

    display: block;

    margin-top: 10px;

    color: #007bff;

    text-decoration: none;

    font-size: 14px;

  }

  .forgot-password-link:hover {

    text-decoration: underline;

  }

//src\app\forgot-password\forgot-password.component.html

<div class="forgot-password-container">

  <h2>Forgot Password</h2>

  <label>Email:

    <input type="email" [(ngModel)]="email" required />

  </label>

  <button (click)="sendPassword()">Send Password</button>

</div>

//src\app\forgot-password\forgot-password.component.spec.ts

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { ForgotPasswordComponent } from './forgot-password.component';

describe('ForgotPasswordComponent', () => {

  let component: ForgotPasswordComponent;

  let fixture: ComponentFixture<ForgotPasswordComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [ForgotPasswordComponent]

    })

    .compileComponents();

    fixture = TestBed.createComponent(ForgotPasswordComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

//src\app\forgot-password\forgot-password.component.ts

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

import { HttpClient } from '@angular/common/http';

import { Router } from '@angular/router';

import { ToastrService } from 'ngx-toastr';

import { CommonModule } from '@angular/common';

import { FormsModule } from '@angular/forms';

@Component({

  selector: 'app-forgot-password',

  standalone: true,

  imports: [CommonModule, FormsModule],

  templateUrl: './forgot-password.component.html',

  styleUrl: './forgot-password.component.css'

})

export class ForgotPasswordComponent {

  email: string = '';

  constructor(

    private http: HttpClient,

    private router: Router,

    private toastr: ToastrService

  ) {}

  // Called when user submits the form

  sendPassword(): void {

this.http.post('http://localhost:8080/api/users/forgot-password', { email: this.email })

      .subscribe({

        next: () => {

          this.toastr.success('Password sent to your email!');

          this.router.navigate(['/login']);

        },

        error: () => {

          this.toastr.error('Email not found!');

        }

});

  }

}

//src\app\landing\landing.component.css

.landing-container {

    display: flex;

    flex-direction: column;

    align-items: center;

    justify-content: center;

    padding: 50px;

    height: 100vh;

    background-color: #f0f8ff;

  }

  h1 {

    color: #333;

    font-size: 36px;

  }

  p {

    color: #666;

    font-size: 18px;

  }

  .landing-buttons {

    margin-top: 30px;

  }

  button {

    background-color: #007bff;

    color: white;

    border: none;

    margin: 0 10px;

    padding: 12px 24px;

    font-size: 16px;

    cursor: pointer;

    border-radius: 8px;

    transition: background-color 0.3s ease;

  }

  button:hover {

    background-color: #0056b3;

  }

//src\app\landing\landing.component.html

<div class="landing-container">

    <h1>Welcome to Travel Booking System</h1>

    <p>Book your next adventure or manage your travel packages easily.</p>

    <div class="landing-buttons">

      <a  routerLink="/register"><button>Register</button></a>

      <a  routerLink="/login"><button>Login</button></a>

    </div>

  </div>

//src\app\landing\landing.component.spec.ts

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { LandingComponent } from './landing.component';

describe('LandingComponent', () => {

  let component: LandingComponent;

  let fixture: ComponentFixture<LandingComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [LandingComponent]

    })

    .compileComponents();

    fixture = TestBed.createComponent(LandingComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

//src\app\landing\landing.component.ts

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

import { RouterLink } from '@angular/router';

@Component({

  selector: 'app-landing',

  standalone: true,

  imports: [RouterLink], // Enable routerLink directive in template

  templateUrl: './landing.component.html',

  styleUrl: './landing.component.css'

})

export class LandingComponent {}

//src\app\login\login.component.css

.login-container {

    max-width: 400px;

    margin: 50px auto;

    padding: 20px 30px;

    border: 1px solid #ccc;

    border-radius: 8px;

    box-shadow: 0 4px 8px rgba(0,0,0,0.1);

    background-color: #fff;

    font-family: 'Segoe UI', sans-serif;

  }

  .login-container h2 {

    text-align: center;

    color: #2c3e50;

    margin-bottom: 20px;

  }

  .login-container label {

    display: block;

    margin-top: 15px;

    font-weight: 600;

  }

  .login-container input {

    width: 100%;

    padding: 8px 10px;

    margin-top: 5px;

    border: 1px solid #aaa;

    border-radius: 4px;

    outline-color: #007bff;

  }

  .login-container button {

    width: 100%;

    padding: 10px;

    background-color: #007bff;

    color: #fff;

    font-weight: bold;

    border: none;

    margin-top: 20px;

    border-radius: 4px;

    cursor: pointer;

  }

  .login-container button:hover {

    background-color: #0056b3;

  }

  .forgot-password-wrapper {

    margin-top: 12px;

    text-align: center;

  }

  .forgot-password-link {

    color: #007bff;

    text-decoration: underline;

    cursor: pointer;

  }

//src\app\login\login.component.html

<div class="login-container">

    <h2>Login</h2>

    <form (ngSubmit)="login()">

      <label>Email:</label>

  <input type="email" [(ngModel)]="credentials.email" name="email" required />

      <label>Password:</label>

      <input type="password" [(ngModel)]="credentials.password" name="password" required />

      <button type="submit">Login</button>

    </form>

//src\app\login\login.component.spec.ts

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { LoginComponent } from './login.component';

describe('LoginComponent', () => {

  let component: LoginComponent;

  let fixture: ComponentFixture<LoginComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [LoginComponent]

    })

    .compileComponents();

    fixture = TestBed.createComponent(LoginComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

//src\app\login\login.component.ts

interface LoginResponse {

  token: string;

  role: string;

  id: number;

}

import { FormsModule } from '@angular/forms';

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

import { Router } from '@angular/router';

import { RouterModule } from '@angular/router';

import { HttpClient } from '@angular/common/http';

import { ToastrService } from 'ngx-toastr';

@Component({

  selector: 'app-login',

  standalone: true,

  imports: [FormsModule, RouterModule],

  templateUrl: './login.component.html',

  styleUrls: ['./login.component.css']

})

export class LoginComponent {

  credentials = {

    email: '',

    password: ''

  };

  constructor(

    private http: HttpClient,

    private router: Router,

    private toastr: ToastrService

  ) {}

  login() {

this.http.post<LoginResponse>('http://localhost:8080/api/auth/login', this.credentials)

      .subscribe({

        next: (res) => {

          this.toastr.success('✅ Login successful');

          const role = res.role;

          const token = res.token;

          const id = res.id;

          // Save token to localStorage

          localStorage.setItem('token', token);

          localStorage.setItem('role', role);

          localStorage.setItem('userId', id.toString());

          // Redirect based on role

          if (role === 'ADMIN') {

            this.router.navigate(['/admin-dashboard']);

          } else if (role === 'AGENT') {

            this.router.navigate(['/agent-dashboard']);

          } else if (role === 'CUSTOMER') {

            this.router.navigate(['/customer-dashboard']);

          }

        },

        error: () => {

          this.toastr.error('❌ Invalid credentials');

        }

      });

  }

  navigateToForgotPassword(event: Event) {

    event.preventDefault();

    this.router.navigate(['/forgot-password']);

  }

}

   <!-- Outside the <form> -->

<div class="forgot-password-wrapper">

  <a [routerLink]="['/forgot-password']" class="forgot-password-link">Forgot Password?</a>

</div>

  </div>

//src\app\login\login.component.css

.login-container {

    max-width: 400px;

    margin: 50px auto;

    padding: 20px 30px;

    border: 1px solid #ccc;

    border-radius: 8px;

    box-shadow: 0 4px 8px rgba(0,0,0,0.1);

    background-color: #fff;

    font-family: 'Segoe UI', sans-serif;

  }

  .login-container h2 {

    text-align: center;

    color: #2c3e50;

    margin-bottom: 20px;

  }

  .login-container label {

    display: block;

    margin-top: 15px;

    font-weight: 600;

  }

  .login-container input {

    width: 100%;

    padding: 8px 10px;

    margin-top: 5px;

    border: 1px solid #aaa;

    border-radius: 4px;

    outline-color: #007bff;

  }

  .login-container button {

    width: 100%;

    padding: 10px;

    background-color: #007bff;

    color: #fff;

    font-weight: bold;

    border: none;

    margin-top: 20px;

    border-radius: 4px;

    cursor: pointer;

  }

  .login-container button:hover {

    background-color: #0056b3;

  }

  .forgot-password-wrapper {

    margin-top: 12px;

    text-align: center;

  }

  .forgot-password-link {

    color: #007bff;

    text-decoration: underline;

    cursor: pointer;

  }

//src\app\login\login.component.html

<div class="login-container">

    <h2>Login</h2>

    <form (ngSubmit)="login()">

      <label>Email:</label>

  <input type="email" [(ngModel)]="credentials.email" name="email" required />

      <label>Password:</label>

      <input type="password" [(ngModel)]="credentials.password" name="password" required />

      <button type="submit">Login</button>

    </form>

   <!-- Outside the <form> -->

<div class="forgot-password-wrapper">

  <a [routerLink]="['/forgot-password']" class="forgot-password-link">Forgot Password?</a>

</div>

  </div>

//src\app\login\login.component.spec.ts

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { LoginComponent } from './login.component';

describe('LoginComponent', () => {

  let component: LoginComponent;

  let fixture: ComponentFixture<LoginComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [LoginComponent]

    })

    .compileComponents();

    fixture = TestBed.createComponent(LoginComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

//src\app\login\login.component.ts

interface LoginResponse {

  token: string;

  role: string;

  id: number;

}

import { FormsModule } from '@angular/forms';

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

import { Router } from '@angular/router';

import { RouterModule } from '@angular/router';

import { HttpClient } from '@angular/common/http';

import { ToastrService } from 'ngx-toastr';

@Component({

  selector: 'app-login',

  standalone: true,

  imports: [FormsModule, RouterModule],

  templateUrl: './login.component.html',

  styleUrls: ['./login.component.css']

})

export class LoginComponent {

  credentials = {

    email: '',

    password: ''

  };

  constructor(

    private http: HttpClient,

    private router: Router,

    private toastr: ToastrService

  ) {}

  login() {

this.http.post<LoginResponse>('http://localhost:8080/api/auth/login', this.credentials)

      .subscribe({

        next: (res) => {

          this.toastr.success('✅ Login successful');

          const role = res.role;

          const token = res.token;

          const id = res.id;

          // Save token to localStorage

          localStorage.setItem('token', token);

          localStorage.setItem('role', role);

          localStorage.setItem('userId', id.toString());

          // Redirect based on role

          if (role === 'ADMIN') {

            this.router.navigate(['/admin-dashboard']);

          } else if (role === 'AGENT') {

            this.router.navigate(['/agent-dashboard']);

          } else if (role === 'CUSTOMER') {

            this.router.navigate(['/customer-dashboard']);

          }

        },

        error: () => {

          this.toastr.error('❌ Invalid credentials');

        }

      });

  }

  navigateToForgotPassword(event: Event) {

    event.preventDefault();

    this.router.navigate(['/forgot-password']);

  }

}

//src\app\my-profile\my-profile.component.css

/\* Center the profile content \*/

h2 {

    text-align: center;

    margin-top: 40px;

    font-size: 26px;

    font-weight: bold;

    color: #333;

  }

  p {

    text-align: center;

    font-size: 16px;

    margin: 10px 0;

    color: #444;

  }

  /\* Button container \*/

  .profile-buttons {

    display: flex;

    justify-content: center;

    gap: 15px;

    margin-top: 30px;

  }

  /\* Style the buttons \*/

  .profile-buttons button {

    padding: 10px 20px;

    font-size: 14px;

    font-weight: bold;

    border: none;

    border-radius: 6px;

    cursor: pointer;

    transition: background-color 0.3s ease;

  }

  /\* Different colors for each button \*/

  .profile-buttons button:nth-child(1) {

    background-color: #007bff;

    color: white;

  }

  .profile-buttons button:nth-child(2) {

    background-color: #dc3545;

    color: white;

  }

  .profile-buttons button:nth-child(3) {

    background-color: #6c757d;

    color: white;

  }

  /\* Hover effects \*/

  .profile-buttons button:hover {

    opacity: 0.9;

  }

//src\app\my-profile\my-profile.component.html

<div class="profile-card">

    <h2>My Profile</h2>

    <p><strong>Name: </strong>{{ user.name }}</p>

    <p><strong>Email: </strong>{{ user.email }}</p>

    <p><strong>Contact Number:</strong> {{ user.contactNumber }}</p>

    <p><strong>Role:</strong> {{ user.role }}</p>

    <div class="profile-buttons">

      <button (click)="goToUpdate()">Update Profile</button>

      <button (click)="deleteProfile()">Delete Profile</button>

      <button (click)="logout()">Logout</button>

    </div>

  </div>

//src\app\my-profile\my-profile.component.spec.ts

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { MyProfileComponent } from './my-profile.component';

describe('MyProfileComponent', () => {

  let component: MyProfileComponent;

  let fixture: ComponentFixture<MyProfileComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [MyProfileComponent]

    })

    .compileComponents();

    fixture = TestBed.createComponent(MyProfileComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

//src\app\my-profile\my-profile.component.ts

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

import { FormsModule } from '@angular/forms';

import { HttpClient } from '@angular/common/http';

import { Router } from '@angular/router';

import { ToastrService } from 'ngx-toastr';

@Component({

  standalone: true,

  selector: 'app-my-profile',

  imports: [FormsModule],

  templateUrl: './my-profile.component.html',

  styleUrls: ['./my-profile.component.css']

})

export class MyProfileComponent implements OnInit {

  user: any = {};

  constructor(private http: HttpClient, private router: Router, private toastr: ToastrService) {}

  ngOnInit(): void {

    this.loadProfile();

  }

  loadProfile() {

    const token = localStorage.getItem('token');

this.http.get('http://localhost:8080/api/users/myprofile', {

      headers: { Authorization: `Bearer ${token}` }

    }).subscribe(res => this.user = res);

  }

  updateProfile() {

    const token = localStorage.getItem('token');

this.http.put('http://localhost:8080/api/users/myprofile', this.user, {

      headers: { Authorization: `Bearer ${token}` }

    }).subscribe({

      next: () => this.toastr.success('Profile updated'),

      error: () => this.toastr.error('Update failed')

    });

  }

  deleteProfile() {

    const token = localStorage.getItem('token');

this.http.delete('http://localhost:8080/api/users/myprofile', {

      headers: { Authorization: `Bearer ${token}` }

    }).subscribe({

      next: () => {

        localStorage.clear();

        this.toastr.success('Profile deleted');

        this.router.navigate(['/']);

      },

      error: () => this.toastr.error('Delete failed')

    });

  }

  goToUpdate(){

    this.router.navigate(['/update-profile']);

  }

  logout() {

    localStorage.clear();

    this.toastr.success('Logged out');

    this.router.navigate(['/']);

  }

}

//src\app\view-package\view-package.component.css

//src\app\view-package\view-package.component.html

<div \*ngIf="packageData">

    <h2>{{ packageData.title }}</h2>

    <p><strong>Description:</strong> {{ packageData.description }}</p>

    <p><strong>Duration:</strong> {{ packageData.duration }} days</p>

    <p><strong>Price:</strong> ₹{{ packageData.price }}</p>

    <p><strong>Max Capacity:</strong> {{ packageData.maxCapacity }}</p>

    <p><strong>Country:</strong> {{ packageData.country }}</p>

    <p><strong>Destination:</strong> {{ packageData.destination }}</p>

    <p><strong>Trip Type:</strong> {{ packageData.tripType }}</p>

    <p><strong>Start Date:</strong> {{ packageData.tripStartDate }}</p>

    <p><strong>End Date:</strong> {{ packageData.tripEndDate }}</p>

    <h3>Offer:</h3>

    <p>Coupon Code: {{ packageData.offer?.couponCode }}</p>

    <p>Description: {{ packageData.offer?.description }}</p>

    <p>Discount: {{ packageData.offer?.discountPercentage }}%</p>

    <h3>Flights</h3>

    <ul>

        <li \*ngFor="let f of packageData.flights">

            <strong>{{ f.airline }}</strong><br>

            From: {{ f.fromCity }} → To: {{ f.toCity }}<br>

            Departure: {{ f.departureTime }} | Arrival: {{ f.arrivalTime }}

        </li>

    </ul>

    <h3>Hotels</h3>

    <ul>

        <li \*ngFor="let h of packageData.hotels">

            {{ h.name }} - {{ h.city }} ({{ h.rating }}★)<br>

            Nights: {{ h.nights }}<br>

            Cost per Night: ₹{{ h.costPerNight }}<br>

            Total Cost: ₹{{ h.nights \* h.costPerNight }}

        </li>

    </ul>

    <h3>Sightseeing</h3>

    <ul>

      <li \*ngFor="let s of packageData.sightseeing">

        {{ s.location }} - {{ s.description }}

      </li>

    </ul>

    <h3>Itinerary</h3>

    <ul>

      <li \*ngFor="let i of packageData.itinerary">

        Day {{ i.dayNumber }}: {{ i.activityTitle }} - {{ i.activityDescription }}

      </li>

    </ul>

    <br />

    <button (click)="router.navigate(['/agent-dashboard'])">Back to Dashboard</button>

</div>

//src\app\view-package\view-package.component.spec.ts

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { ViewPackageComponent } from './view-package.component';

describe('ViewPackageComponent', () => {

  let component: ViewPackageComponent;

  let fixture: ComponentFixture<ViewPackageComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [ViewPackageComponent]

    })

    .compileComponents();

    fixture = TestBed.createComponent(ViewPackageComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

//src\app\view-package\view-package.component.ts

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

import { ActivatedRoute, Router } from '@angular/router';

import { HttpClient, HttpClientModule } from '@angular/common/http';

import { CommonModule } from '@angular/common';

import { FormsModule } from '@angular/forms';

import { RouterModule } from '@angular/router';

@Component({

  selector: 'app-view-package',

  standalone: true,

  imports: [CommonModule, FormsModule, HttpClientModule, RouterModule],

  templateUrl: './view-package.component.html',

  styleUrl: './view-package.component.css'

})

export class ViewPackageComponent implements OnInit {

  packageId!: number;

  packageData: any;

  constructor(private route: ActivatedRoute, private http: HttpClient, public router: Router){}

  ngOnInit(): void {

    this.packageId = Number(this.route.snapshot.paramMap.get('id'));

    const url = `http://localhost:8080/api/packages/${this.packageId}`;

    this.http.get<any>(url).subscribe({

      next: (res) =>{

        this.packageData = res.data;

        console.log("Loaded packages:", this.packageData);

      },

      error: (err) =>{

        console.error('Error fetching package', err);

        alert('Failed to load package details');

      }

    });

  }

}

//src\app\app.component.html

<router-outlet></router-outlet>

//src\app\app.component.ts

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

import { RouterOutlet } from '@angular/router';

@Component({

  selector: 'app-root',

  imports: [RouterOutlet],

  templateUrl: './app.component.html',

  styleUrl: './app.component.css'

})

export class AppComponent {

  title = 'travel-booking-system-frontend';

}

//src\app\app.routes.ts

// Import Routes type from Angular Router

import { Routes } from '@angular/router';

import { LoginComponent } from './login/login.component';

import { LandingComponent } from './landing/landing.component';

import { RegisterComponent } from './register/register.component';

import { AdminUserManageComponent } from './admin-user-manage/admin-user-manage.component';

export const routes: Routes = [

  { path: '', component: LandingComponent },

  { path: 'register', component: RegisterComponent },

  { path: 'login', component: LoginComponent },

  // Dashboard placeholders

  { path: 'admin-dashboard', loadComponent: () => import('./admin-dashboard/admin-dashboard.component').then(m => m.AdminDashboardComponent) },

  { path: 'agent-dashboard', loadComponent: () => import('./agent-dashboard/agent-dashboard.component').then(m => m.AgentDashboardComponent) },

  { path: 'customer-dashboard', loadComponent: () => import('./customer-dashboard/customer-dashboard.component').then(m => m.CustomerDashboardComponent) },

  { path: 'my-profile', loadComponent: () => import('./my-profile/my-profile.component').then(m => m.MyProfileComponent) },

  { path: 'admin/users', loadComponent: () => import('./admin-user-manage/admin-user-manage.component').then(m => m.AdminUserManageComponent) },

  { path: 'update-profile', loadComponent: () => import('./update-profile/update-profile.component').then(m => m.UpdateProfileComponent) },

  { path: 'admin-user-manage' , loadComponent: () => import('./admin-user-manage/admin-user-manage.component').then(m =>AdminUserManageComponent)},

  { path: 'admin/agent-packages/:id', loadComponent: () => import('./admin-agent-packages/admin-agent-packages.component').then(m => m.AdminAgentPackagesComponent)},

  { path: 'forgot-password', loadComponent: () => import('./forgot-password/forgot-password.component').then(m => m.ForgotPasswordComponent)},

  //For package editing/updating in agent dashboard

  { path: 'edit-package/:id', loadComponent:() => import('./edit-package/edit-package.component').then(m => m.EditPackageComponent)},

  //For viewing the details of the package by the agent(retrieve)

  { path: 'view-package/:id', loadComponent:() => import('./view-package/view-package.component').then(m => m.ViewPackageComponent)}

];

//src\app\auth-expired.interceptor.spec.ts

import { TestBed } from '@angular/core/testing';

import { AuthExpiredInterceptor } from './auth-expired.interceptor';

import { Router } from '@angular/router';

import { ToastrService } from 'ngx-toastr';

describe('AuthExpiredInterceptor', () => {

  let interceptor: AuthExpiredInterceptor;

  beforeEach(() => {

    TestBed.configureTestingModule({

      providers: [

        AuthExpiredInterceptor,

        { provide: Router, useValue: { navigate: jasmine.createSpy() } },

        { provide: ToastrService, useValue: { warning: jasmine.createSpy() } }

      ]

    });

    interceptor = TestBed.inject(AuthExpiredInterceptor);

  });

  it('should be created', () => {

    expect(interceptor).toBeTruthy();

  });

});

//src\app\auth-expired.interceptor.ts

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

import {

  HttpInterceptor, HttpRequest, HttpHandler, HttpEvent, HttpErrorResponse

} from '@angular/common/http';

import { Observable, throwError } from 'rxjs';

import { catchError } from 'rxjs/operators';

import { Router } from '@angular/router';

import { ToastrService } from 'ngx-toastr';

@Injectable()

export class AuthExpiredInterceptor implements HttpInterceptor {

  constructor(private router: Router, private toastr: ToastrService) {}

  intercept(req: HttpRequest<any>, next: HttpHandler): Observable<HttpEvent<any>> {

    const token = localStorage.getItem('token');

    const authReq = token

      ? req.clone({

          headers: req.headers.set('Authorization', `Bearer ${token}`)

        })

      : req;

    return next.handle(authReq).pipe(

      catchError((error: HttpErrorResponse) => {

        if (error.status === 401 || error.status === 403) {

          this.toastr.warning('Session expired. Please log in again.');

          localStorage.clear();

          this.router.navigate(['/login']);

        }

        return throwError(() => error);

      })

    );

  }

}

//src\index.html

<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <title>TravelBookingSystemFrontend</title>

  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="icon" type="image/x-icon" href="favicon.ico">

</head>

<body>

  <app-root></app-root>

</body>

</html>

//src\main.ts

import { bootstrapApplication } from '@angular/platform-browser';

import { AppComponent } from './app/app.component';

// Required to provide routing

import { provideRouter } from '@angular/router';

import { routes } from './app/app.routes';

// Import standard modules for forms and HTTP

import { importProvidersFrom } from '@angular/core';

import { FormsModule } from '@angular/forms';

import { HTTP\_INTERCEPTORS, HttpClientModule, withInterceptorsFromDi } from '@angular/common/http';

// ✅ Toast + Animations

import { BrowserAnimationsModule } from '@angular/platform-browser/animations';

import { provideToastr } from 'ngx-toastr';  // ✅ Use provideToastr here

//for interceptor for jwt token expiration

import { provideHttpClient } from '@angular/common/http';

import { AuthExpiredInterceptor } from './app/auth-expired.interceptor';

bootstrapApplication(AppComponent, {

  providers: [

    provideHttpClient(withInterceptorsFromDi()),

    provideRouter(routes),

    {

      provide: HTTP\_INTERCEPTORS,

      useClass: AuthExpiredInterceptor,

      multi: true

    },

    // ✅ Correct usage of importProvidersFrom for standard modules

    importProvidersFrom(

      FormsModule,

      HttpClientModule,

      BrowserAnimationsModule

    ),

    // ✅ provideToastr must be outside importProvidersFrom

    provideToastr({

      timeOut: 3000,

      positionClass: 'toast-bottom-center',

      preventDuplicates: true

    })

  ]

});