**Angular 9 application => CURD Operation**

1. Download latest version of **node js**
2. Install angular cli latest version -> **npm install –g @angular/cli@latest**
3. Add **helper** folder in app folder then add below files in **helper** folder
4. Jwt.interceptor.ts

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

import { HttpRequest, HttpHandler, HttpEvent, HttpInterceptor } from '@angular/common/http';

import { Observable } from 'rxjs';

import { SessionStorageService } from '../service/session-storage.service';

@Injectable()

export class JwtInterceptor implements HttpInterceptor {

constructor(private sessionStorageService: SessionStorageService) { }

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

const token = this.sessionStorageService.getAccessToken();

if (request.url.toLowerCase() != "https://localhost:44371/api/authentication/login") {

if (token != null) {

request = request.clone({

setHeaders: {

Authorization: `Bearer ${token}`,

'Content-Type': 'application/json'

}

});

}

}

else {

request = request.clone({

setHeaders: {

'Content-Type': 'application/json'

}

});

}

return next.handle(request);

}

}

====================================================================================

1. Auth.guard.ts

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

import { Router, CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot } from '@angular/router';

import { SessionStorageService } from '../service/session-storage.service';

@Injectable({ providedIn: 'root' })

export class AuthGuard implements CanActivate {

constructor(

private router: Router,

private sessionStorageService: SessionStorageService

) { }

canActivate(route: ActivatedRouteSnapshot, state: RouterStateSnapshot) {

debugger;

const user = this.sessionStorageService.getUserFromSession();

debugger;

if (user != null) {

debugger;

// logged in so return true

return true;

} else {

debugger;

// not logged in so redirect to login page with the return url

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

return false;

}

}

}

===================================================================================

1. Add **service** folder in app folder => add below services in **service** folder

==================================================================================

1. Authentication.service.ts

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

import { Observable, of, throwError } from 'rxjs';

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

import { catchError, tap, map } from 'rxjs/operators';

import { LoginModel } from '../login/models/login.model';

import { SessionStorageService } from '../service/session-storage.service';

import { TokenModel } from '../models/token.model';

import { BehaviorSubject } from 'rxjs';

const apiServer = "https://localhost:44371/api/authentication/";

@Injectable({

providedIn: 'root'

})

export class AuthenticationService {

public serviceTimer: Observable<number>;

public userName: BehaviorSubject<string> = new BehaviorSubject<string>("");

httpOptions = {

headers: new HttpHeaders({

'Content-Type': 'application/json',

})

}

constructor(private http: HttpClient,

private sessionStorageService: SessionStorageService) {

this.serviceTimer = null;

}

setUserName(userEmail: string) {

this.userName.next(userEmail);

}

getUserName() {

return this.userName;

}

loginUser(loginModel: LoginModel): Observable<any> {

const url = apiServer + "login";

return this.http.post<any>(url, JSON.stringify(loginModel)).pipe(

map(response => {

return response;

}),

catchError(error => {

return error;

})

);

}

refreshToken(refreshToken: string): Observable<any> {

const url = apiServer + "refreshtoken";

return this.http.post<any>(url, JSON.stringify(refreshToken)).pipe(

map(response => {

return response;

}),

catchError(error => {

return error;

})

);

}

validateUserAuthentication() {

const session = this.sessionStorageService.getServiceTokenModel();

var accessToken = session.accessToken;

var refreshToken = session.refreshToken;

if (accessToken != null && refreshToken != null) {

return true;

}

else {

return false;

}

}

setTokenFromResponse(data: any) {

const expireDate = new Date(data.accessTokenExpiry);

const utcDate = new Date();

const startTime = Date.UTC(utcDate.getUTCFullYear(), utcDate.getUTCMonth(),

utcDate.getUTCDate(), utcDate.getUTCHours(), utcDate.getUTCMinutes(),

utcDate.getUTCSeconds(), utcDate.getUTCMilliseconds());

const expireTime = Date.UTC(expireDate.getUTCFullYear(), expireDate.getUTCMonth(),

expireDate.getUTCDate(), expireDate.getUTCHours(), expireDate.getUTCMinutes(),

expireDate.getUTCSeconds(), expireDate.getUTCMilliseconds());

const model = new TokenModel();

model.accessTokenExpiryTime = (expireTime - startTime);

model.accessToken = data.accessToken;

model.refreshToken = data.refreshToken;

this.sessionStorageService.setServiceToken(model);

}

}

=================================================================================

1. Session-storage.service.ts

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

import { Observable, of, throwError } from 'rxjs';

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

import { catchError, tap, map } from 'rxjs/operators';

import { LoginModel } from '../login/models/login.model';

import { TokenModel } from '../models/token.model';

const apiServer = "https://localhost:44371/api/authentication/";

@Injectable({

providedIn: 'root'

})

export class SessionStorageService {

httpOptions = {

headers: new HttpHeaders({

'Content-Type': 'application/json',

})

}

constructor(private http: HttpClient) { }

setServiceToken(data: any) {

sessionStorage.setItem("accessToken", data.accessToken);

sessionStorage.setItem("refreshToken", data.refreshToken);

sessionStorage.setItem("accessTokenExpiryTime", data.accessTokenExpiryTime.toString());

}

getAccessToken() {

return sessionStorage.getItem("accessToken");

}

getUserFromSession() {

var user = sessionStorage.getItem("currentUser");

if (user != null) {

return user;

}

return null;

}

getServiceTokenModel() {

const tokenModel = new TokenModel();

tokenModel.accessToken = sessionStorage.getItem("accessToken");

tokenModel.refreshToken = sessionStorage.getItem("refreshToken");

const expireTime = sessionStorage.getItem("accessTokenExpiryTime");

if (expireTime) {

tokenModel.accessTokenExpiryTime = Number.parseInt(expireTime);

}

else {

tokenModel.accessTokenExpiryTime = 300000;

}

return tokenModel;

}

}

==============================================================================

1. Employee.service.ts

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

import { Observable, of, throwError } from 'rxjs';

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

import { catchError, tap, map } from 'rxjs/operators';

import { EmployeeModel } from '../models/employee.model';

const apiServer = "https://localhost:44371/api/employee/";

@Injectable({

providedIn: 'root'

})

export class EmployeeService {

httpOptions = {

headers: new HttpHeaders({

'Content-Type': 'application/json',

})

}

constructor(private http: HttpClient) { }

getEmployeelist(): Observable<any> {

const url = apiServer + "getEmployeeList";

return this.http.get(url).pipe(

map(response => {

return response;

}),

catchError(error => {

return error;

})

);

}

getEmployeeById(id: number): Observable<any> {

const url = apiServer + "getEmployeeById/" + id;

return this.http.get(url).pipe(

map(response => {

return response;

}),

catchError(error => {

return error;

})

);

}

saveEmployee(employee: EmployeeModel): Observable<any> {

const url = apiServer + "saveEmployee";

return this.http.post<any>(url, JSON.stringify(employee)).pipe(

map(response => {

return response;

}),

catchError(error => {

return error;

})

);

}

updateEmployee(employee: EmployeeModel): Observable<any> {

const url = apiServer + "updateEmployee";

return this.http.put<any>(url, JSON.stringify(employee)).pipe(

map(response => {

return response;

}),

catchError(error => {

return error;

})

);

}

deleteEmployee(id: number): Observable<any> {

const url = apiServer + "deleteEmployee/" + id;

return this.http.delete(url).pipe(

map(response => {

return response;

}),

catchError(error => {

return error;

})

);

}

}

================================================================================

1. User.service.ts

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

import { Observable, of, throwError } from 'rxjs';

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

import { catchError, tap, map } from 'rxjs/operators';

const apiServer = "https://localhost:44371/api/user/";

@Injectable({

providedIn: 'root'

})

export class UserService {

httpOptions = {

headers: new HttpHeaders({

'Content-Type': 'application/json',

})

}

constructor(private http: HttpClient) {

}

getUserDetails(userName: string): Observable<any> {

const url = apiServer + "getUserDetails/" + userName;

return this.http.get(url).pipe(

map(response => {

return response;

}),

catchError(error => {

return error;

})

);

}

getUserList(): Observable<any> {

const url = apiServer + "getUserList";

return this.http.get(url).pipe(

map(response => {

return response;

}),

catchError(error => {

return error;

})

);

}

}

=====================================================================================

1. Add **model** folder in **app** folder and add below models in **model** folder
2. Employye.model.ts

export class EmployeeModel {

public empId: number;

public empName: string;

public empEmailId: string;

public empaddress: string;

}

=================================================================================

1. Token.model.ts

export class TokenModel {

public accessToken: string;

public refreshToken: string;

public accessTokenExpiryTime: number;

}

===============================================================================

1. User.model.ts

export class UserInfo {

public userId: number;

public userName: string;

public firstName: string;

public lastName: string;

}

==============================================================================

1. Login.model.ts

export class LoginModel {

public userName: string;

public password: string;

}

=======================================================================================

1. Add **login** **folder** into app folder and add below files in it
2. Login.module.ts

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

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

import { FormsModule } from '@angular/forms'

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

import { AuthenticationService } from '../service/authentication.service';

import { SessionStorageService } from '../service/session-storage.service';

@NgModule({

declarations: [

LoginComponent

],

imports: [

CommonModule,

FormsModule,

],

providers: [AuthenticationService, SessionStorageService],

bootstrap: [LoginComponent]

})

export class LoginModule { }

==================================================================================

1. Login.component.ts

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

import { LoginModel } from '../login/models/login.model';

import { AuthenticationService } from '../service/authentication.service';

import { SessionStorageService } from '../service/session-storage.service';

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

import { UserService } from '../../app/service/user.service';

@Component({

selector: 'app-login',

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

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

})

export class LoginComponent implements OnInit {

loginModel: LoginModel = new LoginModel();

constructor(private authenticationService: AuthenticationService,

private sessionStorageService: SessionStorageService,

private router: Router,

private userService: UserService) { }

ngOnInit(): void {

}

login() {

this.authenticationService.setUserName(this.loginModel.userName);

this.authenticationService.loginUser(this.loginModel).subscribe((response) => {

debugger;

if (response != null) {

this.authenticationService.setTokenFromResponse(response);

this.getUserDetails(this.loginModel.userName);

}

}, (error) => {

console.log(error);

})

}

getUserDetails(emailId: string) {

this.userService.getUserDetails(emailId).subscribe((result: any) => {

sessionStorage.setItem("currentUser", JSON.stringify(result));

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

}, (error) => {

alert(error);

console.log(error);

});

}

}

============================================================================

1. Login.component.html

<table>

<tr>

<td>User Name</td>

<td>

<input type="text" [(ngModel)]="loginModel.userName" />

</td>

</tr>

<tr>

<td>Password</td>

<td>

<input type="text" [(ngModel)]="loginModel.password" />

</td>

</tr>

<tr>

<td></td>

<td>

<button id="btnLogin" (click)="login()">Login</button>

<button id="btnCancel">Cancel</button>

</td>

</tr>

</table>

==================================================================================

1. App.module.ts

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

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

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

import { AppRoutingModule } from './app-routing.module';

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

import { FormsModule } from '@angular/forms'

import { HomeModule } from '../app/home/home.module';

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

import { JwtInterceptor } from '../app/helper/jwt.interceptor';

import { UserService } from '../app/service/user.service';

import { EmployeeService } from '../app/service/employee.service';

import { AuthenticationService } from '../app/service/authentication.service';

@NgModule({

declarations: [

AppComponent,

LoginComponent

],

imports: [

BrowserModule,

FormsModule,

HttpClientModule,

AppRoutingModule,

HomeModule

],

providers: [

AuthenticationService,

EmployeeService,

UserService,

{ provide: HTTP\_INTERCEPTORS, useClass: JwtInterceptor, multi: true },

],

bootstrap: [AppComponent]

})

export class AppModule { }

==================================================================================

1. App.routing.module.ts

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

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

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

import { HomeComponent } from './home/home.component';

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

import { AuthGuard } from '../app/helper/auth.guard';

const routes: Routes = [

{ path: '', component: AppComponent, canActivate: [AuthGuard]},

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

//{ path: 'home', component: HomeComponent, canActivate: [AuthGuard] }

{

path: 'home',

loadChildren: () => import('../app/home/home.module').then(m => m.HomeModule)

}

];

@NgModule({

imports: [RouterModule.forRoot(routes)],

exports: [RouterModule]

})

export class AppRoutingModule { }

=================================================================================

1. App.component.ts

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

import { AuthenticationService } from '../app/service/authentication.service';

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

import { SessionStorageService } from '../app/service/session-storage.service';

import { interval } from 'rxjs';

import { UserService } from '../app/service/user.service';

import { UserInfo } from './models/user.model';

@Component({

selector: 'app-root',

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

styleUrls: ['./app.component.scss']

})

export class AppComponent {

title = 'ClientApp';

constructor(private authenticationService: AuthenticationService,

private router: Router,

private sessionStorageService: SessionStorageService,

private userService: UserService,

) { }

ngOnInit() {

this.getToken();

}

getToken() {

const isAuth = this.authenticationService.validateUserAuthentication();

if (!isAuth) {

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

}

else {

this.authenticationService.getUserName().subscribe((res: any) => {

//this.getUserDetails(res);

this.refreshServiceToken(false);

}, (error) => {

alert(error);

console.log(error);

});

}

}

refreshServiceToken(isInit: boolean) {

const refreshToken = this.sessionStorageService.getServiceTokenModel().refreshToken;

this.authenticationService.refreshToken(refreshToken).subscribe((result: any) => {

if (result) {

this.authenticationService.setTokenFromResponse(result);

if (!isInit) {

this.refreshServiceTokenInIt();

this.router.navigate(["home"]);

}

}

}, (error) => {

alert(error);

console.log(error);

});

}

refreshServiceTokenInIt() {

const expirationTime = this.sessionStorageService.getServiceTokenModel().accessTokenExpiryTime;

if (expirationTime > 0) {

const intervalTime = expirationTime \* 0.8;

if (!this.authenticationService.serviceTimer) {

this.authenticationService.serviceTimer = interval(intervalTime);

this.authenticationService.serviceTimer.subscribe(x => {

debugger;

console.log('service refresh token called');

this.refreshServiceToken(true);

});

}

}

}

//getUserDetails(emailId: string) {

// this.userService.getUserDetails(emailId).subscribe((result: any) => {

// sessionStorage.setItem("currentUser", JSON.stringify(result));

// this.router.navigate(['/home/employee']);

// }, (error) => {

// alert(error);

// console.log(error);

// });

//}

}

================================================================================

1. App.component.html

<router-outlet></router-outlet>

===============================================================================

1. Add **home** folder in app folder and add below files in it

=============================================================================

1. [Home.module.ts](http://Home.module.ts)

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

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

import { FormsModule } from '@angular/forms'

import { HomeComponent } from '../home/home.component';

import { HomeRoutingModule } from '../../app/home/home-routing.module';

@NgModule({

declarations: [

HomeComponent

],

imports: [

CommonModule,

FormsModule,

HomeRoutingModule,

],

providers: [

],

bootstrap: [HomeComponent]

})

export class HomeModule { }

============================================================================

1. Home-routing.module.ts

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

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

import { HomeComponent } from '../home/home.component';

const routes: Routes = [

{

path: '',

component: HomeComponent,

children: [

{

path: '',

redirectTo: 'employee',

pathMatch:'full'

},

{

path: 'employee',

loadChildren: () => import('../home/employee/employee.module').then(m => m.EmployeeModule)

},

{

path: 'userlist',

loadChildren: () => import('../home/user/user.module').then(m => m.UserModule)

}

]

},

];

@NgModule({

imports: [RouterModule.forChild(routes)],

exports: [RouterModule]

})

export class HomeRoutingModule { }

===========================================================================

1. [Home.comoponent.ts](http://Home.comoponent.ts)

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

@Component({

selector: 'app-home',

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

styleUrls: ['./home.component.scss']

})

export class HomeComponent implements OnInit {

constructor() { }

ngOnInit(): void {

}

}

==========================================================================

1. [Home.component.html](http://Home.component.html)

<div class="row">

<div class="row col-md-12">

<div class="col-md-2" style="background:#0041b3">

<ul>

<li style="list-style-type: none;">

<a style="color:white;" routerLink="/employee">Employee</a>

</li>

</ul>

<ul>

<li style="list-style-type: none;">

<a style="color:white;" routerLink="/userlist">User List</a>

</li>

</ul>

</div>

<div class="col-md-10">

<router-outlet></router-outlet>

</div>

</div>

</div>

================================================================================

1. Add **employee** folder in app -> home folder and below files in it

============================================================================

1. Employee.module.ts

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

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

import { FormsModule } from '@angular/forms'

import { EmployeeComponent } from '../../home/employee/employee.component';

import { EmployeeRoutingModule } from '../employee/employee-routing.module';

@NgModule({

declarations: [

EmployeeComponent

],

imports: [

CommonModule,

FormsModule,

EmployeeRoutingModule

],

providers: [],

bootstrap: [EmployeeComponent]

})

export class EmployeeModule { }

============================================================================

1. Employee-routing.module.ts

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

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

import { EmployeeComponent } from './employee.component';

const routes: Routes = [

{ path: '', component: EmployeeComponent },

];

@NgModule({

imports: [RouterModule.forChild(routes)],

exports: [RouterModule]

})

export class EmployeeRoutingModule { }

=============================================================================

1. Employee.component.ts

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

import { EmployeeService } from '../../service/employee.service';

import { EmployeeModel } from '../../models/employee.model';

@Component({

selector: 'app-employee',

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

styleUrls: ['./employee.component.scss']

})

export class EmployeeComponent implements OnInit {

public empList: EmployeeModel[] = [];

public employee: EmployeeModel = new EmployeeModel();

public isCreateView: boolean = false;

constructor(private employeeService: EmployeeService) { }

ngOnInit(): void {

this.getEmployeeList();

}

getEmployeeList() {

this.isCreateView = false;

this.employeeService.getEmployeelist().subscribe((res: any) => {

this.empList = res;

}, (error) => {

console.log(error);

});

}

createEmployee() {

this.isCreateView = true;

this.employee = new EmployeeModel();

}

editData(emp: any) {

this.isCreateView = true;

this.employee = new EmployeeModel();

this.employee = emp;

}

saveEmployee() {

if (this.employee.empId == undefined || this.employee.empId <= 0) {

this.employee.empId = 0;

this.addEmployee(this.employee);

}

else {

this.updateEmployee(this.employee);

}

}

addEmployee(employee: EmployeeModel) {

this.employeeService.saveEmployee(employee).subscribe((res: any) => {

if (res > 0) {

alert("Record Saved");

}

this.getEmployeeList();

this.isCreateView = false;

}, (error) => {

console.log(error);

});

}

updateEmployee(employee: EmployeeModel) {

this.employeeService.updateEmployee(employee).subscribe((res: any) => {

if (res > 0) {

alert("Record Updated");

}

this.getEmployeeList();

this.isCreateView = false;

}, (error) => {

console.log(error);

});

}

deleteEmployee(emp: EmployeeModel) {

this.isCreateView = false;

this.employeeService.deleteEmployee(emp.empId).subscribe((res: any) => {

if (res > 0) {

alert("Record Deleted");

}

this.getEmployeeList();

}, (error) => {

console.log(error);

});

}

cancel() {

this.isCreateView = false;

}

}

=============================================================================

1. Employee.component.html

<div class="col-md-12" \*ngIf="!isCreateView">

<div class="col-md-12">

<button type="button" name="btnCreate" (click)="createEmployee()">Create</button>

</div>

<div class="col-md-12">

<table>

<thead>

<tr>

<th>

Action

</th>

<th>Employee Name</th>

<th>Email Id</th>

<th>Address</th>

</tr>

</thead>

<tbody>

<tr \*ngFor="let emp of empList" (click)="editData(emp)">

<td>

<button (click)="deleteEmployee(emp)">Delete</button>

</td>

<td>**{{**emp.empName**}}**</td>

<td>**{{**emp.empEmailId**}}**</td>

<td>**{{**emp.empaddress**}}**</td>

</tr>

</tbody>

</table>

</div>

</div>

<div class="col-md-12" \*ngIf="isCreateView">

<form name="employeeForm">

<div class="col-md-12">

<div class="col-md-4">

<span>Employee Name</span>

</div>

<div class="col-md-4">

<input type="text" id="empname" name="empname"

[(ngModel)]="employee.empName" />

</div>

</div>

<div class="col-md-12">

<div class="col-md-4">

<span>Email Id</span>

</div>

<div class="col-md-4">

<input type="text" id="empEmailId" name="empEmailId"

[(ngModel)]="employee.empEmailId" />

</div>

</div>

<div class="col-md-12">

<div class="col-md-4">

<span>Address</span>

</div>

<div class="col-md-4">

<input type="text" id="empAddress" name="empAddress"

[(ngModel)]="employee.empaddress" />

</div>

</div>

<div class="col-md-12">

<button id="btnSave" (click)="saveEmployee()">Save</button>

<button id="btnSave" (click)="cancel()">Cancel</button>

</div>

</form>

</div>

============================================================================

1. Add **user** **folder** in app->home folder and add below files in it

============================================================================

1. User.module.ts

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

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

import { FormsModule } from '@angular/forms'

import { UserRoutingModule } from '../user/user-routing.module';

import { UserComponent } from '../user/user.component';

@NgModule({

declarations: [

UserComponent,

],

imports: [

CommonModule,

FormsModule,

UserRoutingModule

],

providers: [

//EmployeeService

],

bootstrap: [UserComponent]

})

export class UserModule { }

============================================================================

1. User-routing.module.ts

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

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

import { UserComponent } from '../user/user.component';

const routes: Routes = [

{ path: '', component: UserComponent },

];

@NgModule({

imports: [RouterModule.forChild(routes)],

exports: [RouterModule]

})

export class UserRoutingModule { }

=================================================================================

1. User.component.ts

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

import { UserInfo } from '../../models/user.model';

import { UserService } from '../../service/user.service';

@Component({

selector: 'app-user',

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

styleUrls: ['./user.component.scss']

})

export class UserComponent implements OnInit {

public userList: UserInfo[] = [];

constructor(private userService: UserService) { }

ngOnInit(): void {

this.getEmployeeList();

}

getEmployeeList() {

this.userService.getUserList().subscribe((res: any) => {

this.userList = res;

}, (error) => {

console.log(error);

});

}

}

=================================================================================

1. User.component.html

<div class="col-md-12">

<table>

<thead>

<tr>

<th>

Action

</th>

<th>User Name</th>

<th>First Name</th>

<th>Last Name</th>

</tr>

</thead>

<tbody>

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

<td>

<button>Delete</button>

</td>

<td>**{{**user.userName**}}**</td>

<td>**{{**user.firstName**}}**</td>

<td>**{{**user.lastName**}}**</td>

</tr>

</tbody>

</table>

</div>

=================================================================================