# Name :- Sarthak Pagar

# Roll No. :- 40

# Class :- TE(IT)

# Practical 2c :- Create an Angular application which will do following actions: Register User, Login User, Show User Data on Profile Component

**src/index.html**

<!doctype html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>UserAuthApp</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 { enableProdMode } from '@angular/core';

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

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

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

import { routes } from './app/app.routes'; // Make sure this file exists

import { environment } from './environments/environment';

if (environment.production) {

enableProdMode();

}

bootstrapApplication(AppComponent, {

providers: [provideRouter(routes)]

}).catch(err => console.error(err));

**src/environments/environment.ts**

export const environment = {

production: false

};

**src/app/app.component.html**

<div style="text-align:center;">

<h1>

Welcome to the User Authentication App!

</h1>

<nav>

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

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

<a routerLink="/profile">Profile</a>

</nav>

<router-outlet></router-outlet> <!-- Renders routed components -->

</div>

**src/app/app.component.ts**

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

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

@Component({

selector: 'app-root',

standalone: true,

imports: [RouterModule],

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

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

})

export class AppComponent {

title = 'user-auth-app';

}

**src/app/app.config.ts**

import { ApplicationConfig, provideZoneChangeDetection } from '@angular/core';

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

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

export const appConfig: ApplicationConfig = {

providers: [provideZoneChangeDetection({ eventCoalescing: true }), provideRouter(routes)]

};

**src/app/app.module.ts**

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

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

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

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

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

import { ProfileComponent } from './components/profile/profile.component';

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

@NgModule({

declarations: [

LoginComponent,

RegisterComponent,

ProfileComponent

],

imports: [

BrowserModule,

AppRoutingModule

],

bootstrap: [AppComponent] // This should work if you are using the AppComponent standalone as part of routing

})

export class AppModule { }

**src/app/app.routes.ts**

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

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

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

import { ProfileComponent } from './components/profile/profile.component';

export const routes: Routes = [

{ path: '', redirectTo: 'login', pathMatch: 'full' },

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

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

{ path: 'profile', component: ProfileComponent }

];

**src/app/app-routing.module.ts**

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

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

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

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

import { ProfileComponent } from './components/profile/profile.component';

const routes: Routes = [

{ path: '', redirectTo: '/login', pathMatch: 'full' },

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

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

{ path: 'profile', component: ProfileComponent },

];

@NgModule({

imports: [RouterModule.forRoot(routes)],

exports: [RouterModule]

})

export class AppRoutingModule { }

**src/app/services/auth.service.ts**

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

@Injectable({

providedIn: 'root'

})

export class AuthService {

private readonly USERS\_KEY = 'registeredUsers';

private readonly CURRENT\_USER\_KEY = 'currentUser';

constructor() {}

private loadUsers(): Map<string, string> {

const usersJson = localStorage.getItem(this.USERS\_KEY);

return usersJson ? new Map(JSON.parse(usersJson)) : new Map();

}

private saveUsers(users: Map<string, string>) {

localStorage.setItem(this.USERS\_KEY, JSON.stringify(Array.from(users.entries())));

}

register(username: string, password: string): boolean {

const users = this.loadUsers();

if (users.has(username)) {

return false; // Username already taken

}

users.set(username, password);

this.saveUsers(users);

return true;

}

login(username: string, password: string): boolean {

const users = this.loadUsers();

const storedPassword = users.get(username);

if (storedPassword === password) {

localStorage.setItem(this.CURRENT\_USER\_KEY, JSON.stringify({ username }));

return true;

}

return false;

}

getUser() {

const userJson = localStorage.getItem(this.CURRENT\_USER\_KEY);

return userJson ? JSON.parse(userJson) : null;

}

logout() {

localStorage.removeItem(this.CURRENT\_USER\_KEY);

}

}

**src/app/components/login/login.component.ts**

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

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

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

import { AuthService } from '../../services/auth.service';

@Component({

selector: 'app-login',

standalone: true,

imports: [CommonModule, FormsModule],

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

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

})

export class LoginComponent {

username: string = '';

password: string = '';

message: string = '';

constructor(private authService: AuthService) {}

login() {

if (this.authService.login(this.username, this.password)) {

this.message = 'Login successful';

} else {

this.message = 'Invalid credentials';

}

}

}

**src/app/components/login/login.component.html**

<div>

<h2>Login</h2>

<form (ngSubmit)="login()">

<label for="username">Username:</label>

<input type="text" id="username" [(ngModel)]="username" name="username" required>

<label for="password">Password:</label>

<input type="password" id="password" [(ngModel)]="password" name="password" required>

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

</form>

<p>{{ message }}</p>

</div>

**src/app/components/profile/profile.component.ts**

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

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

import { AuthService } from '../../services/auth.service';

interface User {

username: string;

}

@Component({

selector: 'app-profile',

standalone: true,

imports: [CommonModule],

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

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

})

export class ProfileComponent implements OnInit {

user: any;

constructor(private authService: AuthService) {}

ngOnInit() {

this.user = this.authService.getUser(); // Safe to access now

}

}

**src/app/components/profile/profile.component.html**

<div \*ngIf="user">

<h2>Welcome, {{ user.username }}</h2>

<p>This is your profile page.</p>

</div>

<div \*ngIf="!user">

<p>No user logged in.</p>

</div>

**src/app/components/register/register.component.ts**

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

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

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

import { AuthService } from '../../services/auth.service';

@Component({

selector: 'app-register',

standalone: true,

imports: [CommonModule, FormsModule],

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

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

})

export class RegisterComponent {

username: string = '';

password: string = '';

confirmPassword: string = '';

message: string = '';

constructor(private authService: AuthService) {}

register() {

if (this.password === this.confirmPassword) {

if (this.authService.register(this.username, this.password)) {

this.message = 'Registration successful';

} else {

this.message = 'Username already exists';

}

} else {

this.message = 'Passwords do not match';

}

}

}

**src/app/components/register/register.component.html**

<div>

<h2>Register</h2>

<form (ngSubmit)="register()">

<label for="username">Username:</label>

<input type="text" id="username" [(ngModel)]="username" name="username" required>

<label for="password">Password:</label>

<input type="password" id="password" [(ngModel)]="password" name="password" required>

<label for="confirmPassword">Confirm Password:</label>

<input type="password" id="confirmPassword" [(ngModel)]="confirmPassword" name="confirmPassword" required>

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

</form>

<p>{{ message }}</p>

</div>

**How to start**

Open cmd in the same path where project folder is located

give command - > ng serve

it will give you link of local host

open it in your browser

And you are ready to go

to close - > q + ENTER

**Command Prompt**

Microsoft Windows [Version 10.0.19045.5608]

(c) Microsoft Corporation. All rights reserved.

C:\Users\sspag>cd C:\Users\sspag\Desktop\Practicals\6th sem\WAD Practicals\Assignment\Practical 2C

C:\Users\sspag\Desktop\Practicals\6th sem\WAD Practicals\Assignment\Practical 2C>ng new user-auth-app

Would you like to share pseudonymous usage data about this project with the Angular Team

at Google under Google's Privacy Policy at https://policies.google.com/privacy. For more

details and how to change this setting, see https://angular.dev/cli/analytics.

No

Global setting: disabled

Local setting: No local workspace configuration file.

Effective status: disabled

√ Which stylesheet format would you like to use? CSS [ https://developer.mozilla.org/docs/Web/CSS

]

√ Do you want to enable Server-Side Rendering (SSR) and Static Site Generation (SSG/Prerendering)? No

CREATE user-auth-app/angular.json (2714 bytes)

CREATE user-auth-app/package.json (1043 bytes)

CREATE user-auth-app/README.md (1533 bytes)

CREATE user-auth-app/tsconfig.json (942 bytes)

CREATE user-auth-app/.editorconfig (331 bytes)

CREATE user-auth-app/.gitignore (629 bytes)

CREATE user-auth-app/tsconfig.app.json (439 bytes)

CREATE user-auth-app/tsconfig.spec.json (449 bytes)

CREATE user-auth-app/.vscode/extensions.json (134 bytes)

CREATE user-auth-app/.vscode/launch.json (490 bytes)

CREATE user-auth-app/.vscode/tasks.json (980 bytes)

CREATE user-auth-app/src/main.ts (256 bytes)

CREATE user-auth-app/src/index.html (310 bytes)

CREATE user-auth-app/src/styles.css (81 bytes)

CREATE user-auth-app/src/app/app.component.html (20239 bytes)

CREATE user-auth-app/src/app/app.component.spec.ts (966 bytes)

CREATE user-auth-app/src/app/app.component.ts (301 bytes)

CREATE user-auth-app/src/app/app.component.css (0 bytes)

CREATE user-auth-app/src/app/app.config.ts (318 bytes)

CREATE user-auth-app/src/app/app.routes.ts (80 bytes)

CREATE user-auth-app/public/favicon.ico (15086 bytes)

√ Packages installed successfully.

Successfully initialized git.

C:\Users\sspag\Desktop\Practicals\6th sem\WAD Practicals\Assignment\Practical 2C>cd user-auth-app

C:\Users\sspag\Desktop\Practicals\6th sem\WAD Practicals\Assignment\Practical 2C\user-auth-app>ng generate component components/register

CREATE src/app/components/register/register.component.html (24 bytes)

CREATE src/app/components/register/register.component.spec.ts (629 bytes)

CREATE src/app/components/register/register.component.ts (233 bytes)

CREATE src/app/components/register/register.component.css (0 bytes)

C:\Users\sspag\Desktop\Practicals\6th sem\WAD Practicals\Assignment\Practical 2C\user-auth-app>ng generate component components/login

CREATE src/app/components/login/login.component.html (21 bytes)

CREATE src/app/components/login/login.component.spec.ts (608 bytes)

CREATE src/app/components/login/login.component.ts (221 bytes)

CREATE src/app/components/login/login.component.css (0 bytes)

C:\Users\sspag\Desktop\Practicals\6th sem\WAD Practicals\Assignment\Practical 2C\user-auth-app>ng generate component components/profile

CREATE src/app/components/profile/profile.component.html (23 bytes)

CREATE src/app/components/profile/profile.component.spec.ts (622 bytes)

CREATE src/app/components/profile/profile.component.ts (229 bytes)

CREATE src/app/components/profile/profile.component.css (0 bytes)

C:\Users\sspag\Desktop\Practicals\6th sem\WAD Practicals\Assignment\Practical 2C\user-auth-app>ng generate service services/auth

CREATE src/app/services/auth.service.spec.ts (363 bytes)

CREATE src/app/services/auth.service.ts (142 bytes)

C:\Users\sspag\Desktop\Practicals\6th sem\WAD Practicals\Assignment\Practical 2C\user-auth-app>ng serve

Component HMR has been enabled.

If you encounter application reload issues, you can manually reload the page to bypass HMR and/or disable this feature with the `--no-hmr` command line option.

Please consider reporting any issues you encounter here: https://github.com/angular/angular-cli/issues

Initial chunk files | Names | Raw size

polyfills.js | polyfills | 90.20 kB |

main.js | main | 3.03 kB |

styles.css | styles | 95 bytes |

| Initial total | 93.32 kB

Application bundle generation complete. [9.833 seconds]

Watch mode enabled. Watching for file changes...

NOTE: Raw file sizes do not reflect development server per-request transformations.

➜ Local: http://localhost:4200/

➜ press h + enter to show help

h

Shortcuts

press r + enter to force reload browser

press u + enter to show server url

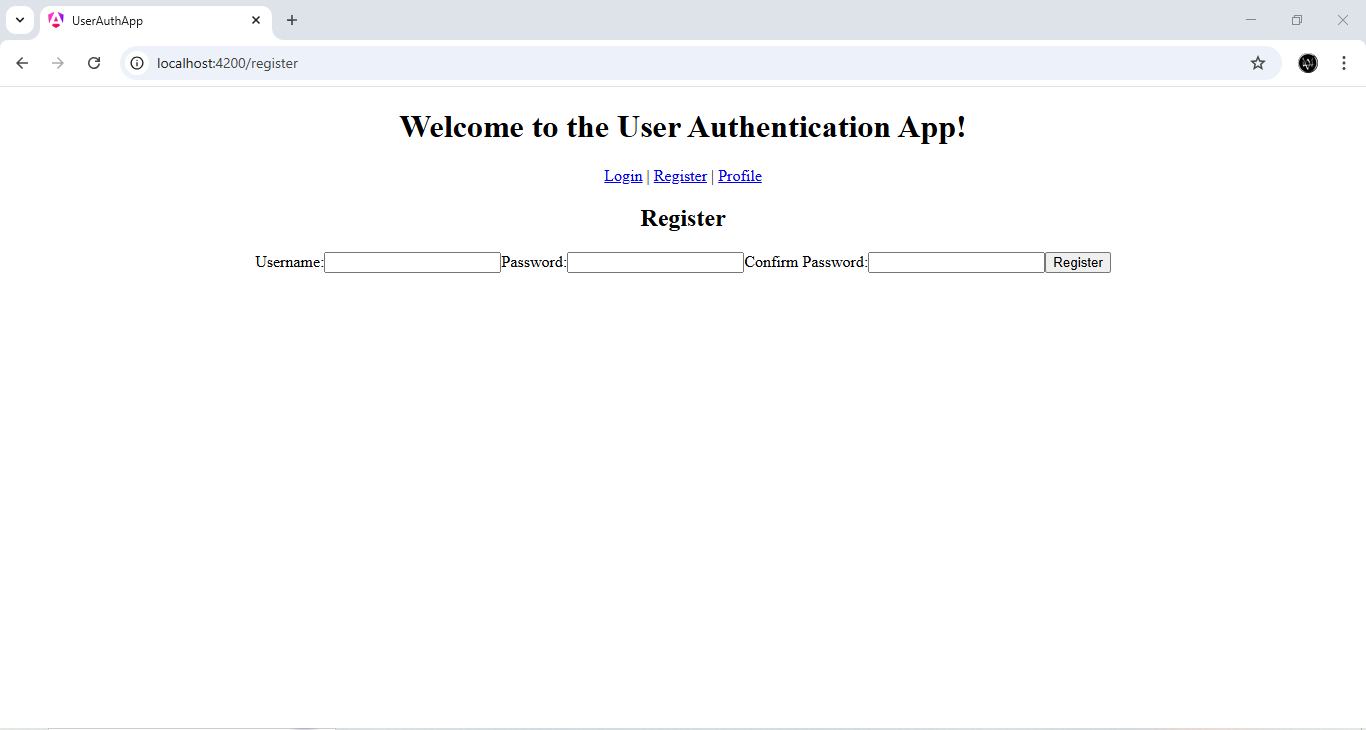
press o + enter to open in browser

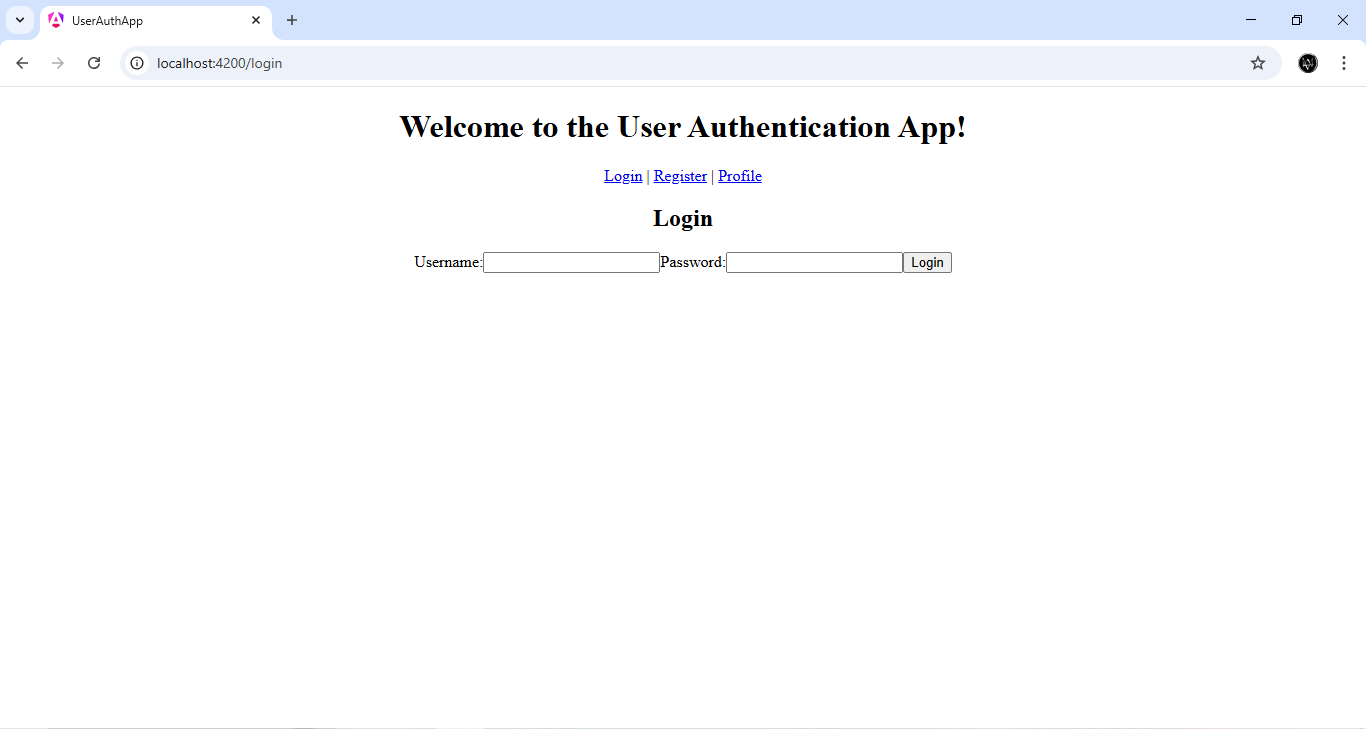
press c + enter to clear console

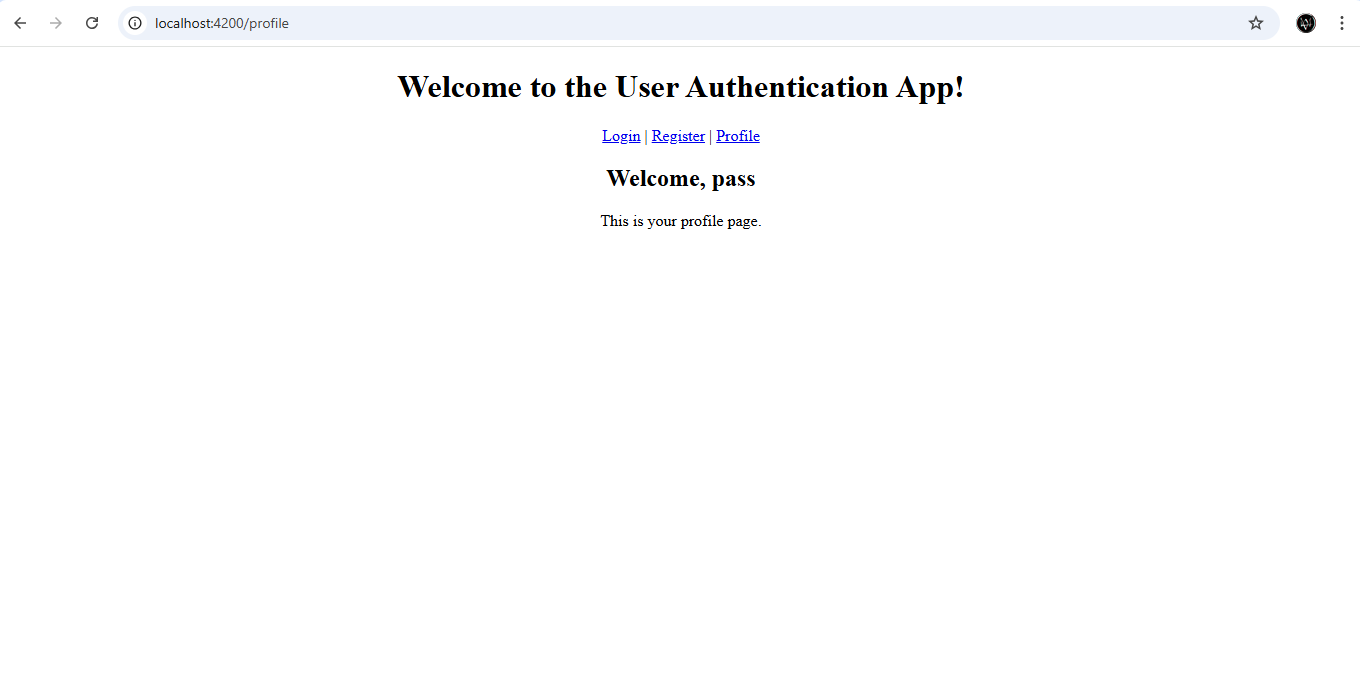
press q + enter to quit

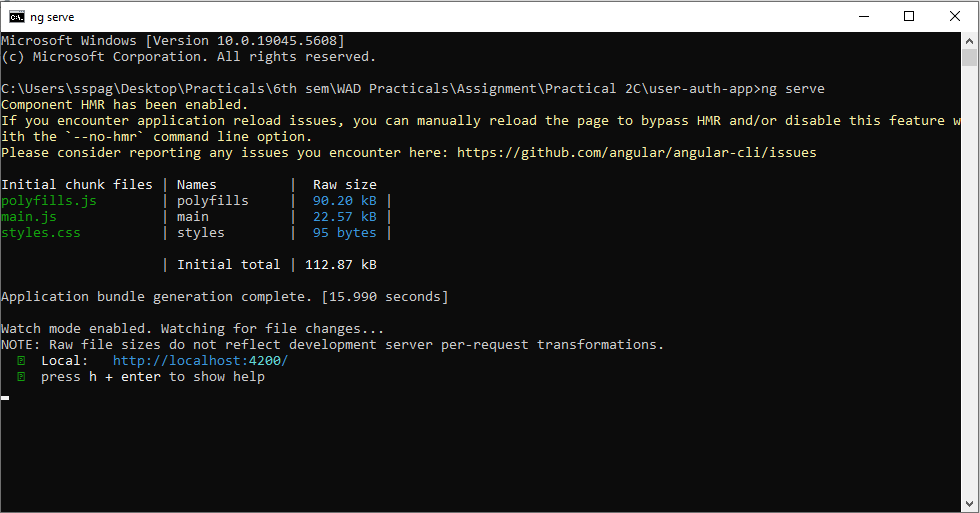
q

**Output :-**

****

****

****

****