build a working **Angular + Node.js authentication app** with:

* Token-based login
* Angular route guards
* Lazy-loaded protected component
* Logout

**PROJECT STRUCTURE OVERVIEW**

angguardserver/

├── auth-server/ ← Node.js backend

│ ├── server.js

│ └── package.json

└── auth-app/ ← Angular frontend

├── src/

└── angular.json

**STEP 1: BACKEND SETUP (Node.js)**

**1.1 Create the folder**

mkdir auth-server && cd auth-server

npm init -y

npm install express cors jsonwebtoken body-parser

**1.2 Create server.js**

// auth-server/server.js

const express = require('express');

const cors = require('cors');

const jwt = require('jsonwebtoken');

const bodyParser = require('body-parser');

const app = express();

const PORT = 3000;

const SECRET = 'super-secret-key';

app.use(cors());

app.use(bodyParser.json());

const USER = { email: 'test@example.com', password: '123456' };

app.post('/login', (req, res) => {

const { email, password } = req.body;

if (email === USER.email && password === USER.password) {

const token = jwt.sign({ email }, SECRET, { expiresIn: '1h' });

res.json({ token });

} else {

res.status(401).json({ message: 'Invalid credentials' });

}

});

function authenticateToken(req, res, next) {

const token = req.headers['authorization']?.split(' ')[1];

if (!token) return res.sendStatus(401);

jwt.verify(token, SECRET, (err, user) => {

if (err) return res.sendStatus(403);

req.user = user;

next();

});

}

app.get('/protected-data', authenticateToken, (req, res) => {

res.json({ message: 'You accessed protected data!', user: req.user });

});

app.listen(PORT, () => console.log(`Server running at http://localhost:${PORT}`));

**1.3 Run it**

node server.js

**STEP 2: ANGULAR APP SETUP**

**2.1 Create Angular project**

cd ..

ng new auth-app --routing --standalone

cd auth-app

**2.2 Install needed packages**

npm install

**STEP 3: CREATE COMPONENTS**

**3.1 Generate everything:**

ng g component app --standalone # AppComponent

ng g component home --standalone # HomeComponent

ng g component pages/login --standalone

ng g component pages/protected —standalone

**3.2 Create Auth Service**

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

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

@Injectable({ providedIn: 'root' })

export class AuthService {

private loggedIn = signal(!!localStorage.getItem('token'));

login(token: string) {

localStorage.setItem('token', token);

this.loggedIn.set(true);

}

logout() {

localStorage.removeItem('token');

this.loggedIn.set(false);

}

isLoggedIn() {

return this.loggedIn();

}

getToken() {

return localStorage.getItem('token') || '';

}

}

**STEP 4: CREATE ROUTES AND GUARDS**

**4.1 Route Guard**

**src/app/guards/auth.guard.ts**

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

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

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

export const authGuard = () => {

const auth = inject(AuthService);

const router = inject(Router);

return auth.isLoggedIn() || router.createUrlTree(['/login']);

};

**4.2 Routes**

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

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

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

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

import { authGuard } from './guards/auth.guard';

export const routes: Routes = [

{ path: '', component: HomeComponent },

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

{

path: 'protected',

canActivate: [authGuard],

loadComponent: () =>

import('./pages/protected/protected.component').then(m => m.ProtectedComponent)

},

];

**STEP 5: BUILD COMPONENTS**

**5.1 AppComponent**

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

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

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

@Component({

selector: 'app-root',

standalone: true,

imports: [RouterOutlet],

template: `<router-outlet></router-outlet>`

})

export class AppComponent {}

**5.2 HomeComponent**

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

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

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

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

@Component({

selector: 'app-home',

standalone: true,

imports: [RouterLink, RouterOutlet],

template: `

<h1>Home</h1>

<nav>

<a routerLink="/">Home</a> |

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

<a routerLink="/protected">Protected</a>

</nav>

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

`

})

export class HomeComponent {

auth: AuthService = inject(AuthService);

router: Router = inject(Router);

logout() {

this.auth.logout();

this.router.navigateByUrl('/login');

}

}

**5.3 LoginComponent**

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

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

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

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

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

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

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

@Component({

selector: 'app-login',

standalone: true,

imports: [CommonModule, FormsModule],

template: `

<h2>Login</h2>

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

<input [(ngModel)]="email" name="email" placeholder="Email" required />

<input [(ngModel)]="password" name="password" type="password" placeholder="Password" required />

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

</form>

<p \*ngIf="error" style="color:red">{{ error }}</p>

`

})

export class LoginComponent {

email = '';

password = '';

error = '';

auth = inject(AuthService);

router = inject(Router);

http = inject(HttpClient);

login() {

this.http.post<{ token: string }>('http://localhost:3000/login', {

email: this.email,

password: this.password

}).subscribe({

next: (res) => {

this.auth.login(res.token);

this.router.navigateByUrl('/protected');

},

error: () => this.error = 'Invalid credentials'

});

}

}

**5.4 ProtectedComponent**

**src/app/pages/protected/protected.component.ts**

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

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

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

@Component({

selector: 'app-protected',

standalone: true,

template: `

<h1>Protected Page</h1>

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

`

})

export class ProtectedComponent {

message = '';

http = inject(HttpClient);

auth = inject(AuthService);

ngOnInit() {

const headers = new HttpHeaders().set(

'Authorization',

`Bearer ${this.auth.getToken()}`

);

this.http.get<{ message: string }>('http://localhost:3000/protected-data', { headers })

.subscribe({

next: res => this.message = res.message,

error: () => this.message = 'Access denied'

});

}

}

**STEP 6: Final Setup**

**src/main.ts**

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

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

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

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

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

bootstrapApplication(AppComponent, {

providers: [provideRouter(routes), provideHttpClient()]

});

**src/index.html**

<body>

<app-root></app-root>

</body>

**RUN THIS**

1. Run backend:

cd auth-server

node server.js

1. Run frontend (another terminal)

cd auth-app

ng serve

1. Visit: http://localhost:4200

* Login with:
  + Email: test@example.com
  + Password: 123456
* You’ll be redirected to the protected page.
* Click logout → you’ll go back to login.