📘 Documentație Tehnică - Birthday Reminder

📋 Cuprins

1. Descriere Proiect
2. Tehnologii Utilizate
3. Arhitectura Aplicației
4. Instalare și Configurare
5. Structura Fișierelor
6. Funcționalități Implementate
7. Explicații Cod
8. Testare
9. Barem Punctaj

🎯 Descriere Proiect

**Birthday Reminder** este o aplicație web modernă construită cu Angular care permite utilizatorilor să-și gestioneze lista de prieteni și să primească reminder-uri pentru zilele de naștere importante.

**Scop**: Realizarea unui proiect complet care demonstrează competențe în dezvoltarea Angular, incluzând autentificare, routing, formulare reactive și gestionarea stării.

⚙ Tehnologii Utilizate

Frontend

* **Angular 20.1.6** - Framework principal
* **TypeScript 5.8.2** - Limbaj de programare
* **RxJS 7.8.0** - Programare reactivă
* **HTML5 & CSS3** - Structură și stilizare
* **Reactive Forms** - Formulare cu validări

Backend & Tools

* **JSON Server** - REST API simulat
* **Angular CLI** - Generator de proiecte
* **Concurrently** - Rulare simultană comenzi

🏗 Arhitectura Aplicației

text

src/

├── app/

│ ├── auth/

│ │ ├── login/

│ │ ├── register/

│ │ ├── auth.service.ts

│ │ ├── auth.guard.ts

│ │ └── auth.interceptor.ts

│ ├── features/

│ │ └── birthday-reminder/

│ ├── app.routes.ts

│ ├── app.component.ts

│ └── app.config.ts

├── assets/

├── environments/

└── main.ts

**Arhitectură Standalone Components**: Folițim componente standalone pentru o structură mai modulară și ușor de întreținut.

🚀 Instalare și Configurare

1. Instalare Dependințe

bash

npm install

2. Pornire Aplicație

bash

# Pornire simultană Angular + JSON Server

npm run dev

# Sau separat:

npm run server # JSON Server pe port 3000

ng serve # Angular pe port 4200

3. Build Production

bash

ng build --production

📁 Structura Fișierelor

Fișiere Principale

* **main.ts** - Punctul de intrare al aplicației
* **app.routes.ts** - Configurarea rutelor
* **app.component.ts** - Componenta root

Module Authentication

* **auth.service.ts** - Gestionează logica de autentificare
* **auth.guard.ts** - Protejează rutele private
* **auth.interceptor.ts** - Interceptează request-urile HTTP

Componente

* **login.component.ts** - Formular de autentificare
* **register.component.ts** - Formular de înregistrare
* **birthday-reminder.component.ts** - Panoul principal

⚡ Funcționalități Implementate

✅ Authentication System

* Login cu email/parolă
* Register cu validări complete
* Remember me functionality
* Logout automat

✅ Routing Avansat

* Lazy loading pentru optimizare
* Route protection cu AuthGuard
* Redirectări inteligente

✅ CRUD Operations

* Create - Adăugare prieteni noi
* Read - Afișare listă prieteni
* Update - Editare prieteni existenți
* Delete - Ștergere prieteni cu confirmare

✅ Validări Avansate

typescript

// Exemplu validări complexe

phone: ['', [Validators.required, Validators.pattern(/^07[0-9]{8}$/)]]

✅ Search & Sort

* Căutare în timp real
* Sortare pe multiple coloane
* Filtrare avansată

✅ State Management

* Angular Signals pentru state local
* RxJS BehaviorSubject pentru state global
* Reactive Forms pentru management form

💻 Explicații Cod

1. Authentication Service

typescript

@Injectable({ providedIn: 'root' })

export class AuthService {

private isAuthenticatedSubject = new BehaviorSubject<boolean>(false);

login(email: string, password: string, rememberMe: boolean): Observable<LoginResponse> {

return this.http.post<LoginResponse>(`${this.apiUrl}/login`, { email, password }).pipe(

tap(response => {

// Remember me logic

if (rememberMe) {

localStorage.setItem('authToken', response.token);

} else {

sessionStorage.setItem('authToken', response.token);

}

this.isAuthenticatedSubject.next(true);

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

})

);

}

}

2. Route Guard

typescript

export const authGuard: CanActivateFn = () => {

const authService = inject(AuthService);

const router = inject(Router);

return authService.isAuthenticated().pipe(

take(1),

map(isAuthenticated => {

if (!isAuthenticated) {

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

return false;

}

return true;

})

);

};

3. Formular Reactiv cu Validări

typescript

this.friendForm = this.fb.group({

firstName: ['', [Validators.required, Validators.minLength(2)]],

lastName: ['', [Validators.required, Validators.minLength(2)]],

phone: ['', [Validators.required, Validators.pattern(/^07[0-9]{8}$/)]],

city: ['', [Validators.required]],

birthdate: ['', [Validators.required]]

});

4. Sortare și Căutare

typescript

get sortedFriends() {

return this.filteredFriends.sort((a, b) => {

const fieldA = a[this.sortField()].toString().toLowerCase();

const fieldB = b[this.sortField()].toString().toLowerCase();

return this.sortDirection() === 'asc'

? fieldA.localeCompare(fieldB)

: fieldB.localeCompare(fieldA);

});

}

🧪 Testare

Testare Manuală

1. **Login**: eve.holt@reqres.in / cityslicka
2. **Register**: Date personale valide
3. **CRUD Operations**: Adăugare, editare, ștergere prieteni
4. **Search**: Testare căutare în listă
5. **Sort**: Click pe header-uri pentru sortare

Testare Remember Me

javascript

// Verificare storage

localStorage.getItem('authToken'); // Remember me ON

sessionStorage.getItem('authToken'); // Remember me OFF

📊 Barem Punctaj

✅ Cerințe Îndeplinite (8.5p/10p)

* **Login + Register** (2.5p) - ✅ Compleat
* **Lazy Loading** (0.5p) - ✅ Implementat
* **@Input/@Output + Serviciu** (0.5p) - ✅ Signals + Services
* **Tabel/Listă** (3p) - ✅ CRUD + Search + Sort
* **Signal** (0.5p) - ✅ Angular Signals
* **Cod Curat** (0.5p) - ✅ Structură modulară
* **Aspect Plăcut** (0.5p) - ✅ Design responsive
* **Punct Oficiu** (1p) - ✅ Acordat

🔄 Cerințe Bonus

* **Conectare Backend** (0.5p) - ✅ JSON Server
* **Funcționalitate Extra** (0.5p) - ✅ Remember Me
* **Librărie Extra** (0.5p) - ⏳ Potențial NgZorro

🎯 Concluzie

Proiectul **Birthday Reminder** demonstrează o înțelegere profundă a ecosystemului Angular și a principiilor de dezvoltare modernă. Aplicația oferă o experiență completă utilizatorului de la autentificare până la gestionarea avansată a datelor.

**Puncte Tari**:

* ✅ Arhitectură modulară și scalabilă
* ✅ Validări robuste și feedback utilizator
* ✅ Performance optimizată cu lazy loading
* ✅ UX/UI intuitiv și responsive
* ✅ Cod maintainable și bine documentat

Proiectul poate fi extins ușor cu funcționalități adicionale precum notificări, export date sau integrări cu platforme externe.

**Autor**: [Balca Eduard Andrei]  
**Data**: [31-08-2025]  
**Tehnologii**: Angular 20, TypeScript, RxJS, JSON Server