SOCIALHUB

Applicazione Angular per connettere le idee dei cittadini verso la creazione di un centro urbano migliore

Autrice: Barletta Chiara
GitHub: https://github.com/junior5969/SocialHub

Indice

1.Obiettivo

Creare un'app Angular che, in seguito ad autenticazione via token, consenta di gestire utenti e visualizzare i loro post. Sarà inoltre possibile creare nuovi utenti, nuovi post e commenti per ogni post.

2.Funzionalità

1. Login con Token generato da GoRest.
2. Gestione utenti con ricerca, creazione, eliminazione e dettagli.
3. Gestione post con ricerca e creazione
4. Visualizzazione commenti per post e loro creazione
5. Logout per terminare la sessione.
6. Loader per la gestione dello stato di caricamento.

3.Architettura

Componenti → Homepage, Users, UserDetail, User, Posts, Notfound

La **Homepage** introduce l'applicazione con una call-to-action, mentre il **UsersComponent** mostra la lista degli utenti con funzione di ricerca. Ogni utente ha il proprio **UserComponent**, un pannello espandibile con dati essenziali e azioni rapide (Dettagli ed Elimina). Il **UserDetailsComponent** approfondisce il profilo, includendo post e commenti. La gestione dei contenuti è affidata al **PostsComponent**, con lista e ricerca dei post. Infine, il **NotfoundComponent** gestisce i percorsi errati con la pagina di errore 404.

3.Architettura

Componenti ⟶ Button, Card, EmptyState, Form, Header, SearchBar

Il **ButtonComponent** offre pulsanti personalizzabili con supporto a routerLink o EventEmitter. Il **CardComponent** funge da contenitore grafico per i post, mentre **l'EmptyStateComponent** mostra un messaggio quando una ricerca non restituisce risultati. Con il **FormComponent** viene gestita la creazione di nuovi utenti, post e commenti, e il **SearchBarComponent** consente la ricerca rapida di utenti e post. L'**HeaderComponent** rappresenta la testata principale con login/logout, e per la gestione del caricamento è stato integrato il **LoaderComponent.**

3.Architettura

Service ⟶ API, Auth, Loader, ErrorInterceptor

L'**ApiService** gestisce tutte le chiamate API, mentre l'**AuthService** si occupa di login, logout e token. La sicurezza delle rotte è assicurata dall'**AuthGuard**, mentre l'**AuthInterceptor** inserisce automaticamente il token nelle richieste. Per la gestione degli errori viene utilizzato l'**ErrorInterceptor,** mentre per la gestione del loader viene utilizzato il **LoaderService**, che insieme al **LoaderInterceptor**, mostrano e nascondono lo spinner durante le chiamate HTTP

4.Motivazioni

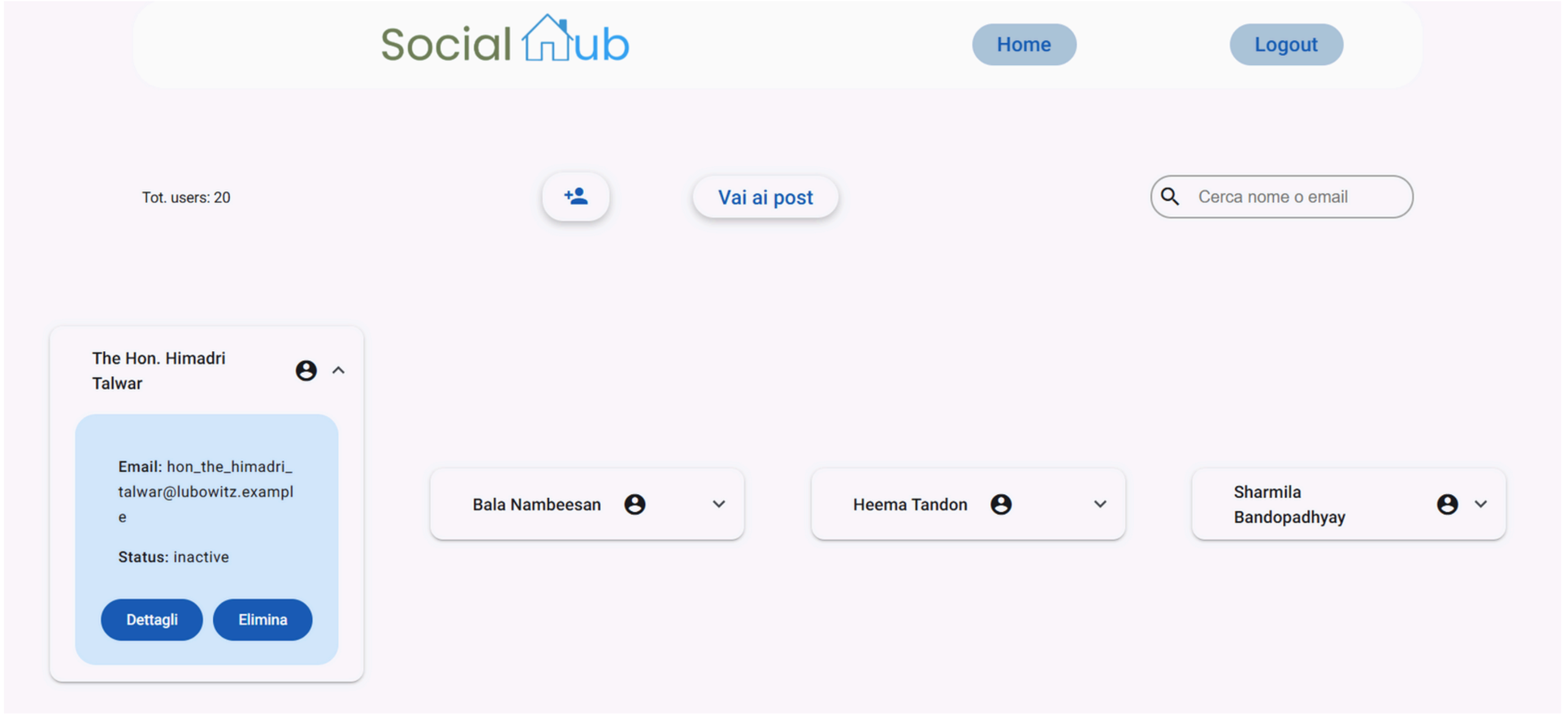
Sono state utilizzate interfacce per utenti, post e commenti per sfruttare al meglio la tipizzazione di TypeScript, ridurre gli errori in fase di sviluppo e rendere il codice più leggibile e manutenibile.

I componenti riutilizzabili sono stati progettati seguendo le best practice di Angular, mentre i servizi centralizzano la logica dell'applicazione.

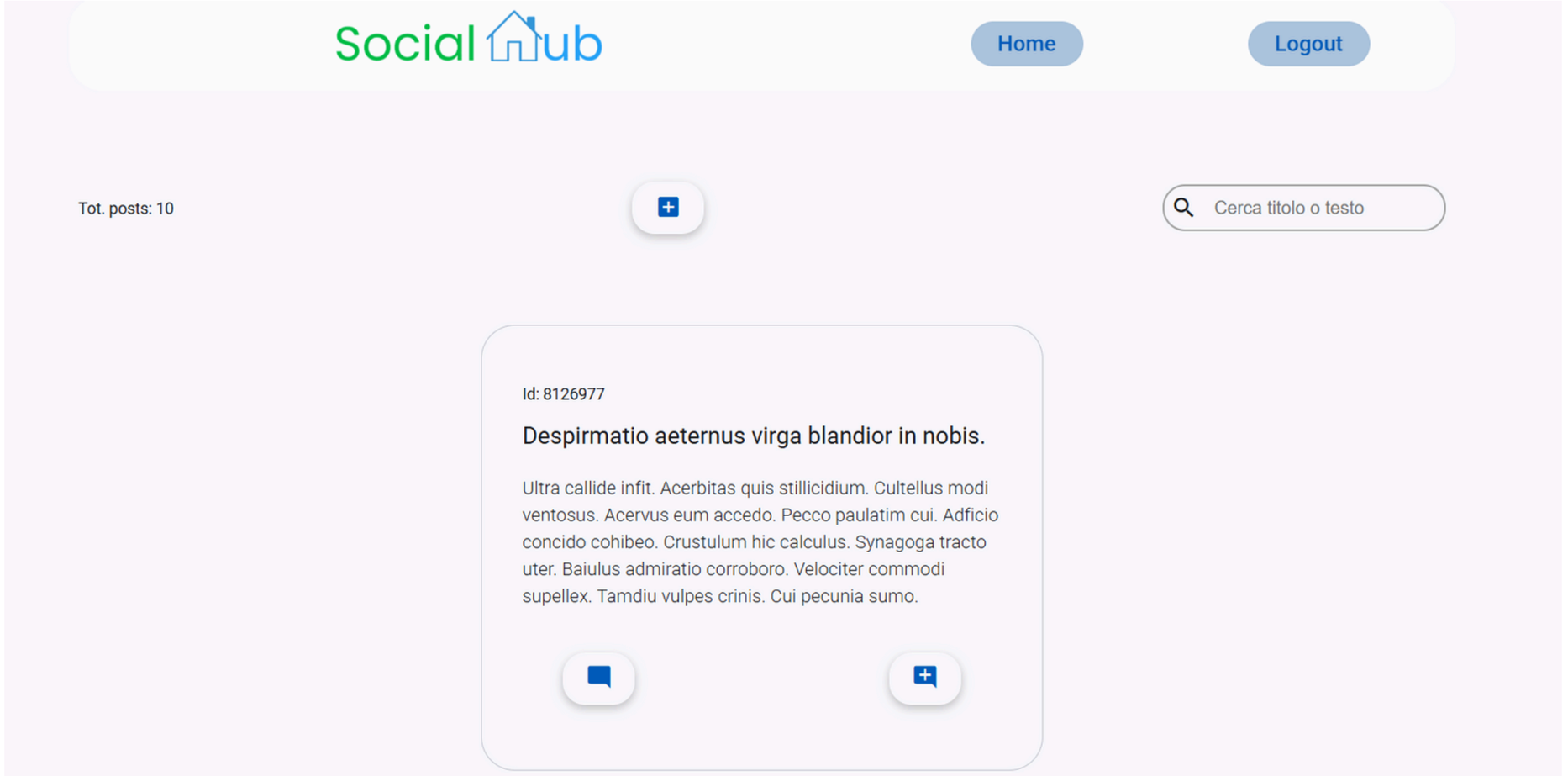
A livello grafico sono stati utilizzati elementi quali un messaggio di ricerca vuota per dare un feedback immediato all'utente, il loader per rendere evidente lo stato delle chiamate API, e un messaggio di errore nel caso di mancata risposta API.

5.Demo (Homepage)

5.Demo (Users)

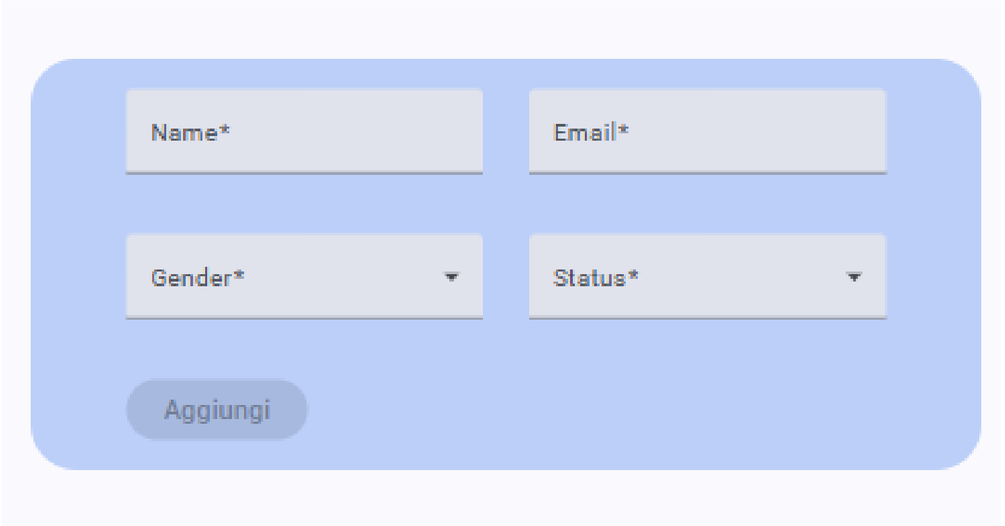

Social Hub
Home
Logout
Tot. users: 20
Vai ai post
Cerca nome o email
The Hon. Himadri Talwar
Email: hon_the_himadri_talwar@lubowitz.example
Status: inactive
Dettagli
Elimina
Bala Nambeesan
Heema Tandon
Sharmila Bandopadhyay

5.Demo (Posts)


Social Hub
Home
Logout
Tot. posts: 10
Cerca titolo o testo
Id: 8126977
Despirmatio aeternus virga blandior in nobis.
Ultra callide infit. Acerbitas quis stillicidium. Cultellus modi ventosus. Acervus eum accedo. Pecco paulatim cui. Adficio concido cohibeo. Crustulum hic calculus. Synagoga tracto uter. Baiulus admiratio corroboro. Velociter commodi supellex. Tamdiu vulpes crinis. Cui pecunia sumo.

5.Demo (Form)

Per aggiungere un nuovo commento



Per aggiungere un nuovo utente