APPLICAZIONI INTERNET

Esercitazione 5 (gio 30/05/2019)

Form e Servizi REST in Angular

(consegna entro domenica 16/6/2019)

Si richiede di estendere l'applicazione front-end, basata su Angular, del laboratorio precedente aggiungendo le seguenti funzionalità:

- Interazione con le API del server per l'interfaccia *presenze* dell'accompagnatore
- Interfacce di registrazione e login con interazione con le API del server

Login

L'interfaccia di login deve soddisfare le specifiche già fornite nelle esercitazioni precedenti. In seguito al login è necessario gestire il token JWT di autorizzazione che viene ritornato dal server e deve essere inviato nell'header delle richieste. A tal fine si faccia riferimento alla guida presente sul sito di Angular University [1] provvedendo a creare un AuthenticationService, a salvare il JWT nel LocalStorage e a inserirlo nelle richieste HTTP tramite un Interceptor.

Registrazione

L'interfaccia di registrazione deve soddisfare le specifiche già fornite nelle esercitazioni precedenti. In aggiunta deve provvedere ad anticipare i controlli già lato client in modo tale che l'operazione di invio del form possa ragionevolmente andare a buon fine. Si suggerisce di provare a implementare lato client anche la verifica che l'indirizzo di email con cui si effettua la registrazione non sia già stato utilizzato. Si indichino opportunamente all'interno della pagina le informazioni richieste, i vincoli e gli errori presenti nel form con suggerimenti per la correzione. Nota: per semplicità si può "appoggiare" il componente che gestisce questa vista sullo stesso AuthenticationService realizzato per la vista di Login.

Presenze

L'interfaccia di gestione delle presenze dei bambini deve dialogare con le API del server per recuperare, per ogni corsa, l'elenco dei bambini prenotati per ciascuna fermata e, in seguito alla registrazione della presenza del bambino, inviare tale informazione al server per archiviarla nella base dati e renderla persistente. Si "appoggi" il componente che gestisce questa vista su un apposito Service che si occupi del collegamento con le API del server.

Nota: i bambini prenotati per ciascuna fermata dovranno essere opportunamente "evidenziati" quando segnati come presenti (rispetto alle specifiche del progetto si aggiunga al fondo della vista, non associato a nessuna fermata, l'elenco dei bambini aderenti al pedibus, ma non prenotati per quella corsa, in modo tale da poterli comunque segnare come presenti se, inaspettatamente, si presentano).

[1] Angular University, "Angular Security - Authentication With JSON Web Tokens (JWT): The Complete Guide", online https://blog.angular-university.io/angular-jwt-authentication/

Indicazioni pratiche

HTTP REST Service

Per l'implementazione di un service per le richieste HTTP si faccia riferimento alle slides 07.Angular-HttpService.pdf e agli esempi disponibili nelle slides e nel repository git del corso (branch *angular-http-service.100-rest-service*) [2].

Routing

Per meglio gestire la presenza di tre viste differenti, si invita a modificare l'applicazione per includere il Routing di Angular per le url login, register e attendance più la creazione delle tre viste (componenti) LoginComponent, RegisterComponent e AttendanceComponent (con i rispettivi file .ts, .html, .css).

Si faccia riferimento alle slides 11.Angular-Routing.pdf e al repository git (branch *angular-router.2019-lab05*) creato come esempio all'indirizzo [3]. In particolare utilizzando il confronto tra i due commit 936438e e 603f749 [4] si possono evidenziare le modifiche da apportare al proprio progetto, soprattutto nella scrittura del file app.module.ts.

[2] https://github.com/internet-applications-at-polito-it/ai19-client-src/tree/angular-http-service.100-rest-service

[3] https://github.com/internet-applications-at-polito-it/ai19-client-src/tree/angular-router.2019-lab05

[4] https://github.com/internet-applications-at-polito-it/ai19-client-src/compare/936438e..603f749

Indicazioni per la consegna

L'elaborato deve essere consegnato come archivio ZIP con nome come segue <nome-gruppo>-lab05-v<N>.zip, dove <nome-gruppo> è il nome, sperabilmente univoco, del gruppo e N è la versione dell'elaborato consegnato.

L’elaborato dovrà essere “impacchettato” utilizzando Docker e Docker Compose. Per l’esecuzione dell’applicazione dovrebbe essere sufficiente eseguire *docker-compose up* nella cartella con il file yml di configurazione (si verifichi la cosa prima della sottomissione). Si eliminino tutti i file non necessari prima dell’upload (es. librerie e altro che può essere reinstallato da maven o npm).

Predisporre un file testuale denominato come README.TXT con:

- lista dei nomi/matricole dei componenti del gruppo
- le informazioni per l’installazione e l’avvio dell’applicazione (es. anche porta HTTP utilizzata)
- le istruzioni per il test delle funzionalità (quali pagine/url aprire, quali informazioni inserire, esempi di interazione con le viste, come fare a testare la registrazione di un nuovo utente)
- IMPORTANTE, si forniscano le credenziali degli utenti da utilizzare per accedere all’applicazione (username e password)

Si forniscano anche più screenshot dell’interfacce realizzate nei loro diversi “stati” di utilizzo (li si numerino in ordine temporale per una consultazione più facile e logica).