Esercizo 1

All'avvio, l'applicazione dovrà automaticamente reindirizzare ad una pagina con url **user** dove sarà fruibile una tabella di utenti aventi le seguenti proprietà:

1. **Nome**
2. **Cognome**
3. **Data di nascita**
4. **Email**

Dovranno essere filtrati, tramite api, gli utenti non cancellati usando la proprietà **isDeleted**

Esercizo 2

Modificare la tabella degli utenti in modo tale che ogni riga abbia un pulsante "Dettagli", al click, il pulsante reindirizzerà l'utente ad una schermata dedicata al dettaglio dell'utente, che dovrà essere visualizzato in una "card" centrata nello schermo

---

Esercizo 3

Modificare la pagina di dettaglio dell'utente affinchè, tramite pulsante "edit" posto nell'angolo in alto a destra nella card sia possibile passare da una modalità "readonly" ad una "edit", infine dovrà essere possibile modificare lo user tramite un pulsante "Salva" oppure annullare le modifiche con un pulsante "Annulla" che farà tornare la card in modalità "readonly" con i dati originali.

Esercizo 4

Modificare la funzionalità di salvataggio appena implementata affinchè l'utente venga reindirizzato alla pagina con vista tabellare dopo il salvataggio avvenuto con successo. Aggiungere un pulsante "Elimina" per ciascuna riga della tabella, l'eliminazione deve avvenire solo dietro conferma dell'utente, terminata l'eliminazione, se questa viene eseguita con successo ricaricare i dati della lista utente affinchè sia aggiornata. L'eliminazione dovrà avvenire in modo logico, controllate le API Postman.

Esercizo 5

Aggiungere un pulsante "Crea utente" che al click reinderizzerà alla pagina di dettaglio dell'utente. La pagina di dettaglio si avvierà in modalità "Edit", l'utente potrà salvare le modifiche dopo aver inserito i dati necessari. Se l'utente preme il pulsante "Annulla" si verrà reindirizzati alla pagina con la tabella degli utenti.

Esercizo 6

Implementare un filtro di ricerca nella pagina con vista tabellare per i campi "Nome", "Cognome" e "Email", la ricerca dovrà avvenire dopo 2500 millisecondi dall'inserimento da parte dell'utente nel campo di testo. Usate gli **observable** per ottenere il comportamento richiesto.

Modificare la chiamata per ottenere la lista degli utenti affinchè restituisca la lista paginata tramite i parametri:

1. *_page*: Number => Default value 1
2. *_per_page*: Number => Default value 10

Integrare il modello dati della lista utenti affinchè sia gestita in modalità paginata e mostri gli utenti restituiti

---

Esercizo 7

Tramite due pulsanti "avanti" e "indietro" posti come footer della tabella dovrà essere possibile navigare tra le pagine degli utenti, inoltre implementare una select affinchè l'utente sia in grado di modificare il numero di elementi per pagina.

Gestire lo stato della paginazione in un servizio dedicato tramite **observable**.

Esercizo 8

Implementare un Interceptor che si occupi di tenere traccia di tutte le attività svolte dall'utente sull'entità "User", dovranno essere salvati tramite api i seguenti tipi di LOG

1. Creazione => *create*
2. Modifica => *update*
3. Eliminazione => *delete*

Esercizo 9

Dalla pagina di dettaglio **user**, tramite pulsante dedicato, si dovrà poter navigare alla pagina dei log relativi a quel **user** esposti come una lista raggruppata per **action**

Implementare una feature che esponga una tabella di tutti i *log*, compresi quelli degli **user** eliminati.
Implementare un filtro che tramite select dia all'utente la possibilità di filtrare i **log** di un determinato user.
