**2. Documento Tecnico – Pagina Dettagli Conto**

**1. Architettura e Struttura del Progetto**

**Tecnologia di Base:**

* **Framework:** Angular (solo front-end).
* **Gestione dei Dati:** Utilizzo di Angular Reactive Forms e binding per visualizzare i dati mock del conto e delle transazioni.
* **Routing:** Configurazione del routing per passare dalla Home Page ai dettagli del conto mediante parametri (es. ID conto).

**2. Componenti e Moduli Angular**

**Componenti Principali:**

* **AccountDetailsComponent:**
  + **Responsabilità:**
    - Gestire la visualizzazione delle informazioni del conto (saldo, dati anagrafici, ecc.).
    - Includere e coordinare la visualizzazione dello storico transazioni.
* **TransactionListComponent:**
  + **Responsabilità:**
    - Visualizzare l’elenco delle transazioni in forma di tabella o lista.
    - Fornire funzionalità di filtro e ordinamento.
* **TransactionDetailComponent (Opzionale):**
  + **Responsabilità:**
    - Visualizzare dettagli aggiuntivi di una singola transazione (in modal o in pagina dedicata).

**3. Struttura del Layout e Routing**

* **Layout della Pagina:**
  + Riutilizzo del HeaderComponent per mantenere la coerenza di navigazione.
  + Sezione principale suddivisa in due pannelli: riepilogo conto e lista transazioni.
* **Routing:**
  + Definizione di una rotta dedicata, es. /account/:id, per caricare il componente AccountDetailsComponent.
  + Possibile utilizzo di parametri per gestire filtri o paginazione dello storico.

**4. Stile e UI**

**Palette e Temi:**

* **Colori:**
  + Sfondo bianco per il contenuto principale.
  + Azzurro (#428fdd) per pulsanti, filtri, link e evidenziazioni.
* **Layout Responsive:**
  + Utilizzo di CSS Grid o Flexbox per adattare la visualizzazione sia su desktop che su dispositivi mobili.
* **Componenti UI:**
  + Utilizzo di componenti di tabelle e cards per la visualizzazione dei dati.
  + Dropdown e input di ricerca per filtri e ordinamenti.

**5. Gestione dei Dati e Interattività**

* **Servizi Angular:**
  + Creazione di un AccountService per simulare il recupero dei dati del conto e dello storico transazioni tramite dati mock.
* **Data Binding e Reactive Forms:**
  + Utilizzo di binding bidirezionale per aggiornare dinamicamente la visualizzazione al variare dei filtri.
* **Gestione degli Errori:**
  + Implementazione di meccanismi per mostrare messaggi di errore in caso di fallimento nel caricamento dei dati.
* **Ottimizzazione delle Prestazioni:**
  + Pagina suddivisa in componenti per migliorare il rendering e la gestione delle liste di transazioni anche in presenza di grandi dataset.

**6. Scalabilità e Manutenibilità**

* **Modularità:**
  + Componenti ben separati per facilitare future modifiche e integrazioni (es. con backend reale).
* **Documentazione Interna:**
  + Commenti e documentazione in codice per facilitare la comprensione e la manutenzione da parte di altri sviluppatori.
* **Estendibilità:**
  + Struttura predisposta per l’inserimento di ulteriori funzionalità, come esportazione avanzata e paginazione.