**2. Documento Tecnico – Creazione di un Bonifico**

**1. Architettura e Struttura del Progetto**

**Tecnologia di Base:**

* **Framework:** Angular (solo front-end).
* **Gestione del Form:** Utilizzo di Angular Reactive Forms per la gestione e la validazione dei campi.
* **Routing:** Configurazione della rotta per accedere alla pagina di bonifico (es. /bonifico).

**2. Componenti e Moduli Angular**

**Componenti Principali:**

* **BonificoComponent (o TransferComponent):**
  + **Responsabilità:**
    - Visualizzare il form per l’inserimento dei dati del bonifico.
    - Gestire la logica di validazione dei campi e il flusso dell’operazione.
  + **Struttura Interna:**
    - Template HTML per il form, suddiviso in sezioni (Beneficiario, Dati Bonifico).
    - File SCSS/CSS per applicare lo stile (palette azzurro e bianco).
    - File TypeScript per la gestione del form, validazioni, e simulazione della transazione.
* **Eventuali Componenti di Validazione:**
  + Componenti o direttive personalizzate per validare il formato IBAN e la correttezza dell’importo.

**3. Struttura del Layout e Routing**

* **Layout della Pagina:**
  + Header comune per tutte le pagine, contenente il logo e il menu.
  + Container centrale che ospita il form di bonifico.
* **Routing:**
  + Configurazione della rotta dedicata al bonifico, es. /bonifico.
  + Implementazione di eventuali guardie di rotta per garantire che solo utenti autenticati possano accedere alla pagina.

**4. Stile e UI**

**Palette e Temi:**

* **Colori:**
  + Utilizzo del bianco come sfondo principale.
  + Elementi interattivi (pulsanti, bordi, messaggi di errore) evidenziati in azzurro (#428fdd).
* **Componenti UI:**
  + Uso di card o container per raggruppare logicamente le sezioni del form.
  + Input design e messaggi di errore integrati per migliorare l’usabilità.

**5. Gestione dei Dati e Interattività**

* **Servizi Angular:**
  + Creazione di un TransferService per simulare l’elaborazione del bonifico e la gestione dei dati mock.
* **Binding e Validazione:**
  + Uso di binding bidirezionale per aggiornare la UI in tempo reale.
  + Implementazione di validazioni tramite Angular Reactive Forms per ogni campo (Validators.required, Validators.pattern per IBAN, ecc.).
* **Feedback e Transizioni:**
  + Visualizzazione di messaggi di conferma o di errore in seguito alla simulazione della transazione, con animazioni CSS per una transizione fluida.

**6. Scalabilità e Manutenibilità**

* **Componentizzazione:**
  + Progettare il componente in maniera modulare per poter eventualmente estendere la funzionalità (es. integrazione con backend reale).
* **Separazione delle Responsabilità:**
  + Distinzione chiara tra la logica di presentazione (componenti) e la logica di business (servizi).
* **Documentazione Interna:**
  + Commenti e linee guida per facilitare la manutenzione e l’eventuale onboarding di nuovi sviluppatori.