1/13/2025

Rishabh Tyagi

NAGARRO SOFTWARE PVT. LTD.

**Insurance Portal**

High Level Design Document

1. **Overview:**

This document outlines the architecture and design for a client-side application for an insurance company. The application will have two Micro Frontends (MFEs) and a container application that facilitates communication between the MFEs. The MFEs will handle tasks such as displaying insurance details and paying premiums. The application will be implemented using **Angular** for all MFEs and the container app.

**Key Requirements:**

1. **Micro Frontend Applications (MFEs):**
   * **Insurance Details MFE:** Displays the user's insurance details.
   * **Premium Payment MFE:** Allows the user to calculate and pay insurance premiums.
2. **Container Application:** Manages the loading of both MFEs and handles cross-communication between them.
3. **Data Storage:** Since there is no backend, **client-side storage** (like **localStorage** or **SessionStorage**) will be used to persist user data.
4. **Cross-Cutting Concerns:**
   * **Web Worker:** Used in the Premium Payment MFE for offloading complex calculations (e.g., discount calculations).
   * **Webpack Bundling:** Used for bundling the applications.
   * **CSS Preprocessor (SCSS):** All styles will be written in SCSS for better structure and maintainability.
   * **Inter-MFE Communication:** The container application will facilitate communication between the MFEs, enabling data sharing (e.g., sending user details from Insurance Details MFE to Premium Payment MFE).
   * **OWASP Top 10 Implementation (CSRF Protection):** Implemented in the forms of both MFEs to protect against CSRF attacks, even though there is no actual backend.
5. **Architecture Diagram:**

Below is a simple architecture diagram for the application setup:

A diagram of a diagram

Description automatically generated

**Detailed Architecture:**

1. **Container Application (Host):**
   * **App Module**: Loads the MFEs dynamically via Module Federation (configured in webpack).
   * **App Routing**: Manages routes for loading the MFEs based on user navigation.
   * **Communication**: Uses **window.postMessage** or shared services to facilitate communication between the MFEs.
2. **Insurance Details MFE (Remote 1):**
   * **Insurance Data Display**: Fetches user insurance data (simulated from localStorage or static data).
   * **Form for User Info**: Includes form fields for the user’s insurance details (e.g., name, policy number, etc.).
3. **Premium Payment MFE (Remote 2):**
   * **Premium Calculator**: Allows the user to calculate premium payments.
   * **Web Worker**: Offloads complex calculations (e.g., premium discounts based on user data).
   * **Payment Form**: Includes fields to input payment details (e.g., amount, method, etc.).
4. **Cross-Cutting Concerns:**

1. **Web Worker:**
   * **Usage:** A worker in the **Premium Payment MFE** is used to offload premium discount calculation.
   * **Implementation:**
     + discount-calculator.worker.ts calculates the premium discount based on input data and returns the result to the main application.
2. **Webpack Bundling:**
   * Each MFE is bundled separately using Webpack.
   * **Module Federation** is used to dynamically load the MFEs into the container app without reloading or recompiling.
3. **CSS Preprocessor:**
   * SCSS is used for all styles to ensure maintainability, structure, and reusability.
   * Shared styles like global themes, color schemes, and font styles are stored in shared/styles.scss.
4. **Inter-MFE Communication:**
   * Communication between MFEs is achieved via the **container application**, which facilitates data sharing.
   * For example, when a user’s insurance details are updated in the **Insurance Details MFE**, they are passed to the **Premium Payment MFE** to adjust the premium calculation.
5. **OWASP Top 10 (CSRF Protection):**
   * CSRF protection is implemented in form submissions. Even though the backend is not real, the frontend will still generate a CSRF token and attach it to form submissions to demonstrate the protection against CSRF attacks.