Application Architecture and Design

1. Architectural Overview

The application follows a **standard three-tier architecture**, separating the user interface, business logic, and data storage. This design promotes maintainability, scalability, and a clear separation of concerns.

- **Frontend (Presentation Layer):** The user interface, built with **Angular 20**, handles all user interactions and displays data. It communicates with the backend via API calls.
- **Backend (Business Logic Layer):** The API, developed with **ASP.NET Core .NET 9**, processes requests from the frontend. It contains the business logic, handles user authentication, and interacts with the database.
- **Database (Data Layer): MS SQL 2022** is used to persist data, including user information and chatbot agent details.

2. Application Flow

The user journey through the application is structured to provide seamless interaction.

- **Public Access (Dashboard):** The public dashboard is the landing page. It is accessible to all users and may display general information or featured agents.
- **Authentication (Sign In/Sign Up):** To create or manage agents, a user must **authenticate**. The Angular frontend directs the user to a sign-in or sign-up page. The backend handles the authentication process, verifying user credentials and issuing a token for subsequent requests.
- **User Dashboard:** After a successful sign-in, the user is navigated to a personalized dashboard. This dashboard displays a list of the user's previously created agents and provides the option to create a new one. All requests to this dashboard are authenticated to ensure data security.
- **Agent Creation:** When a user opts to create a new agent, the frontend sends a request to the backend. The backend's business logic processes this request, creates a new agent entry in the database, and returns a confirmation to the frontend. The user can then immediately begin chatting with the newly created agent.
- **Chat Session:** When a user selects an agent to chat with, the frontend navigates to the chat interface. All chat messages are sent to the backend API, which uses the agent's logic to generate a response. This response is then sent back to the frontend to be displayed in the chat interface.