Project Title: Business Insurance Quote and Policy Management System

1. Overview

The **Business Insurance Quote and Policy Management System** is designed to streamline the generation of insurance quotes, the management of business insurance policies, and the processing of claims, renewals, and customer inquiries. The system will cater to businesses looking to secure insurance for their operations and employees. It will provide easy access to policy creation, claim processing, and quote generation, all while maintaining a user-friendly experience. The application follows **MVC architecture**, making it suitable for both **Java (Spring MVC)** and **.NET (ASP.NET Core MVC)**.

The system will consist of the following modules:

1. **Quote Management**
2. **Policy Management**
3. **Claim Management**
4. **Renewal Management**
5. **Customer Support**

2. Assumptions

- The system will include role-based access, allowing for different user roles such as admin, agent, and customer.
- ORM tools like Hibernate (Java) or Entity Framework (ASP.NET Core) will be used for database interactions.
- The application will support MySQL or SQL Server as the database.
- All user interactions with the application will be through a web-based interface.

3. Module-Level Design

3.1 Quote Management Module

Purpose: Generates and manages business insurance quotes based on user input and policy details.

- **Controller:**
    - QuoteController
        - generateQuote()
        - getQuoteDetails()
        - getAllQuotes()

- **Service:**
    - QuoteService
        - Handles the logic for generating insurance quotes based on business parameters such as business size, coverage, and type of policy.
        - Validates user input and ensures proper calculation of premiums.
- **Model:**
    - **Quote Entity**
        - Attributes:
            - quoteId (PK)
            - businessType
            - coverageAmount
            - premiumAmount
            - validUntil
            - quoteStatus (PENDING, APPROVED, REJECTED)

3.2 Policy Management Module

Purpose: Manages the creation, modification, and retrieval of business insurance policies.

- **Controller:**
    - PolicyController
        - createPolicy()
        - getPolicyDetails()
        - updatePolicy()
        - getAllPolicies()
        - deletePolicy()
- **Service:**
    - PolicyService
        - Manages policy lifecycle, including creation, updates, retrievals, and deletions.
- **Model:**
    - **Policy Entity**
        - Attributes:

        - policyId (PK)
        - policyNumber
        - businessId (FK)
        - coverageAmount
        - policyStartDate
        - policyEndDate
        - policyStatus (ACTIVE, INACTIVE, RENEWED)

3.3 Claim Management Module

Purpose: Handles the claim submission process, including approval, denial, and tracking.

- **Controller:**
    - ClaimController
        - submitClaim()
        - getClaimDetails()
        - updateClaimStatus()
        - getAllClaims()
- **Service:**
    - ClaimService
        - Handles the claim lifecycle, from submission to approval/rejection and tracking of claim status.
- **Model:**
    - **Claim Entity**
        - Attributes:
            - claimId (PK)
            - policyId (FK)
            - claimAmount
            - claimDate
            - claimStatus (OPEN, APPROVED, REJECTED)

3.4 Renewal Management Module

Purpose: Manages the renewal process for business insurance policies, including reminders, updates, and renewal applications.

- **Controller:**
    - RenewalController
        - initiateRenewal()
        - getRenewalStatus()
        - getAllRenewals()
- **Service:**
    - RenewalService
        - Handles policy renewal requests, tracking, and reminders for upcoming renewals.
- **Model:**
    - **Renewal Entity**
        - Attributes:
            - renewalId (PK)
            - policyId (FK)
            - renewalDate
            - newCoverageAmount
            - renewalStatus (PENDING, APPROVED, REJECTED)

3.5 Customer Support Module

Purpose: Provides customer support by allowing businesses to submit inquiries or issues related to their policies or claims.

- **Controller:**
    - SupportController
        - createTicket()
        - getTicketDetails()
        - resolveTicket()
        - getAllTickets()
- **Service:**

- SupportService
  - Manages customer support tickets, including ticket creation, resolution, and ticket status tracking.

- **Model:**
  - **SupportTicket Entity**
    - Attributes:
      - ticketId (PK)
      - userId (FK)
      - issueDescription
      - ticketStatus (OPEN, RESOLVED)
      - createdDate
      - resolvedDate

4. Database Schema

4.1 Table Definitions

1. **Quote Table**

```
CREATE TABLE Quote (
  quoteId INT AUTO_INCREMENT PRIMARY KEY,
  businessType VARCHAR(255),
  coverageAmount DECIMAL(10, 2),
  premiumAmount DECIMAL(10, 2),
  validUntil DATE,
  quoteStatus ENUM('PENDING', 'APPROVED', 'REJECTED')
);
```

2. **Policy Table**

```
CREATE TABLE Policy (
  policyId INT AUTO_INCREMENT PRIMARY KEY,
  policyNumber VARCHAR(50),
  businessId INT,
  coverageAmount DECIMAL(10, 2),
```

```
    policyStartDate DATE,
    policyEndDate DATE,
    policyStatus ENUM('ACTIVE', 'INACTIVE', 'RENEWED')
);
```

3. **Claim Table**

```
CREATE TABLE Claim (
    claimId INT AUTO_INCREMENT PRIMARY KEY,
    policyId INT,
    claimAmount DECIMAL(10, 2),
    claimDate DATE,
    claimStatus ENUM('OPEN', 'APPROVED', 'REJECTED'),
    FOREIGN KEY (policyId) REFERENCES Policy(policyId)
);
```

4. **Renewal Table**

```
CREATE TABLE Renewal (
    renewalId INT AUTO_INCREMENT PRIMARY KEY,
    policyId INT,
    renewalDate DATE,
    newCoverageAmount DECIMAL(10, 2),
    renewalStatus ENUM('PENDING', 'APPROVED', 'REJECTED'),
    FOREIGN KEY (policyId) REFERENCES Policy(policyId)
);
```

5. **SupportTicket Table**

```
CREATE TABLE SupportTicket (
    ticketId INT AUTO_INCREMENT PRIMARY KEY,
    userId INT,
    issueDescription TEXT,
    ticketStatus ENUM('OPEN', 'RESOLVED'),
    createdDate DATE,
    resolvedDate DATE,
    FOREIGN KEY (userId) REFERENCES User(userId)
);
```

6. **User Table**

```
CREATE TABLE User (
    userId INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) UNIQUE,
    password VARCHAR(255),
    email VARCHAR(100),
    role ENUM('ADMIN', 'AGENT', 'CUSTOMER')
);
```

5. Local Deployment Details

1. **Environment Setup:**
    - Ensure MySQL or SQL Server is installed and configured.
    - Install JDK (Java Development Kit) or .NET SDK based on the chosen platform for the application.
    - Set up the database schema using the SQL scripts provided.
2. **Deployment Steps:**
    - Clone the repository from the version control system (GitHub/Bitbucket).
    - Modify application configuration files to set up database connections.
    - Build and deploy the application to the local server (Apache Tomcat for Java or Kestrel for .NET).
    - Test the application locally by accessing the endpoints through a browser or Postman.

6. Conclusion

The **Business Insurance Quote and Policy Management System** provides a robust and scalable solution for managing business insurance operations. This system follows the **MVC architecture** and is designed for both **Java (Spring MVC)** and **.NET (ASP.NET Core MVC)** frameworks. The modular design ensures flexibility and scalability, with key modules including **Quote Management**, **Policy Management**, **Claim Management**, **Renewal Management**, and **Customer Support**. The detailed database schema facilitates efficient storage and processing of data. The system is designed for local deployment with MySQL or SQL Server.