# CyberVault - Broker Portal for Cyber Insurance

By Pragya Singh

## Knowing Cyber Insurance

* Cyber Insurance helps businesses and individuals mitigate risks from cyberattacks, data breaches, and online threats.
* It covers lost revenue, data recovery costs, legal expenses, and other costs incurred during a cyber incident.
* Importance: As cyber threats become more advanced, insurance acts as a financial safety net.
* Example: A stolen company laptop containing sensitive data would result in high costs — Cyber Insurance helps cover those expenses.

## Objective and Scope

* Goal: To streamline the cyber insurance quoting process and enhance risk assessment.
* Target Audience: Insurance brokers and digital users looking for speed and accuracy.
* Key Features: Automated quoting, Risk scoring, Policy comparison.
* Benefits: Improved workflow efficiency and higher accuracy in insurance operations.

## Portal Workflow

* Broker Login: Brokers log in to create quotes for clients.
* Risk Assessment: System evaluates the client’s cyber risk.
* Quote Generation: Portal calculates premiums and compares policies.
* Quote Access: Users can view, edit, soft delete, and compare existing quotes.

## Functional & Technical Features

* Functional Features: Create, edit, submit, and delete quotes, Dashboard with performance metrics, PDF generation, Quote binding, Policy comparison.
* Technical Features: Rating API, Quick quote calculator, Location auto-fetch, SweetAlert integration, Email notifications, Bcrypt password hashing, Validated REST APIs.

## Technology Stack

* Database: SQL Server
* Frontend: Angular 19
* Backend: ASP.NET Web API
* Security: Password encryption using Bcrypt
* PDF Rendering: Using pdf.js

## Design Patterns Used

* Repository Pattern: Decouples business logic and data access.
* Service Pattern: Business logic and communication are separated.
* Interface Segregation: Ensures clean, focused interfaces.
* Layered Architecture: Controller → Service → Repository

## SOLID Principles Followed

* Single Responsibility: Each module serves a single function.
* Interface Segregation: Clean API contracts.
* Open/Closed: Code is extendable without modification.
* Dependency Inversion: Repositories are injected into controllers.

## Out of Scope & Assumptions

* Out of Scope: Multi-user login, Underwriter approvals, State or country-based variations.
* Assumptions: Brokers can only view quotes after binding, Only brokers log in, Rating values are country-agnostic (assumed for one country).

## Future Enhancements

* Multi-user login with underwriter role.
* Payment gateway integration post-binding.
* PDF upload for auto-filling quote data.
* ChatBot for admin-broker group communication.