ASSIGNMENT -1

Insurance Broker Management System. Part I

Project Description:

The **Insurance Broker Management System** allows brokers to manage customer data and insurance policies effectively. The system performs CRUD operations for both customer and policy entities and stores all data in JSON format for easy retrieval and persistence. The application is built using Java Servlets, which interact with the business logic through a well-structured service and repository layer.

Design Decisions:

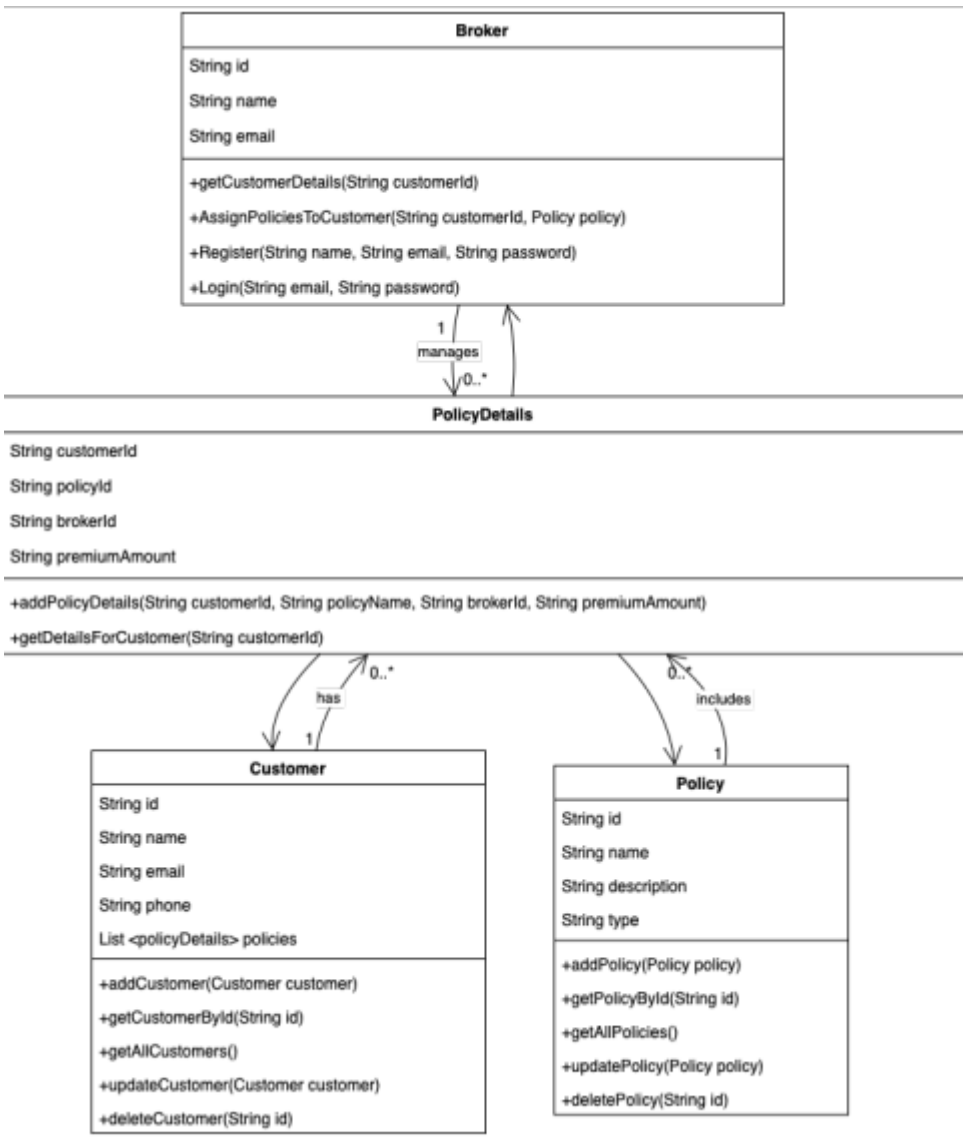
- Architecture: The system follows a standard J2EE Web Application Architecture, with proper separation between Servlet, Service, and Repository layers. This architecture promotes maintainability and scalability.
- File Storage Format: The system uses JSON files for data storage. Both customer and policy data are stored in JSON format, allowing for structured, human-readable data storage.
- Repository Design Pattern: The repository pattern is implemented to separate the logic responsible for accessing the database (or file system) from the business logic. This approach makes the system easier to maintain and test.

Features:

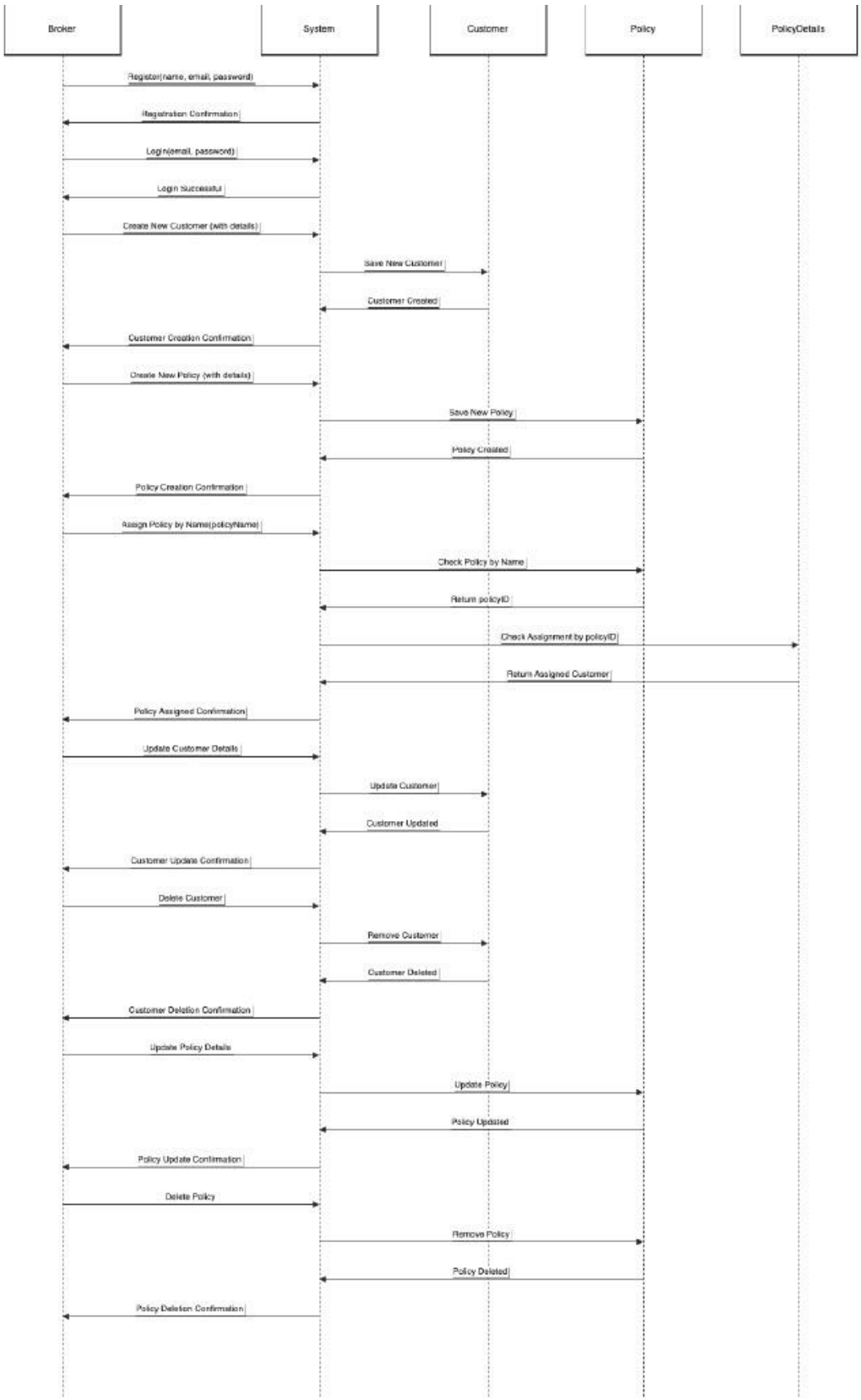
1. Customer Management:
    - Add, update, delete, and view customer data.
    - Customer data is stored in JSON format, and all CRUD operations are managed through the repository pattern.
2. Policy Management:
    - Add, view, and assign policies to customers and brokers.
    - Policies are also stored in JSON format, ensuring consistency with the customer data storage.
3. Thread Safety:
    - The application implements synchronized blocks in servlets to ensure thread safety during concurrent access to shared resources (e.g., customer and policy data).
4. Servlet Mappings:
    - All servlet mappings are done in the web.xml deployment descriptor, adhering to the guidelines of the assignment.

UML Diagrams:

1. Class Diagram:

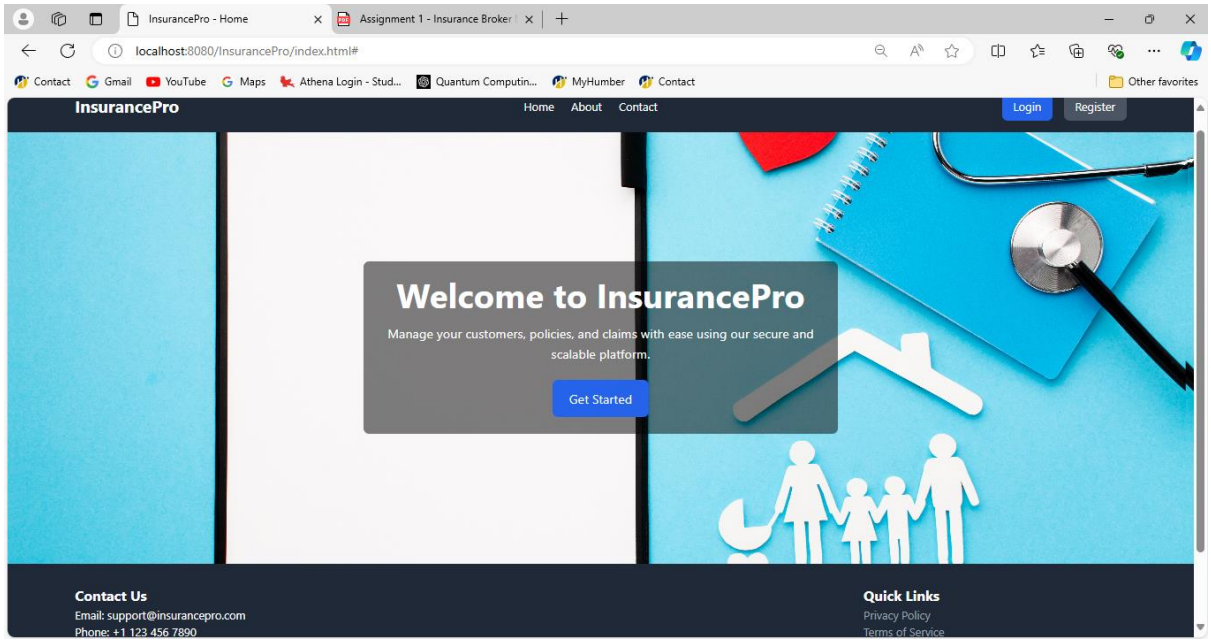


2. Sequence Diagram:

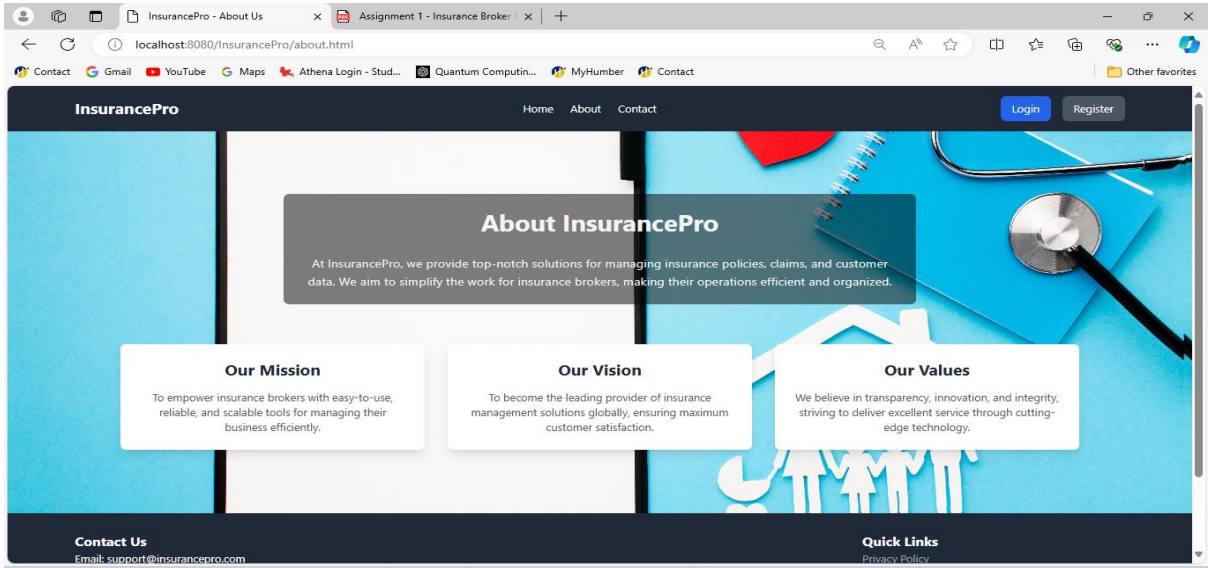


Screenshots:

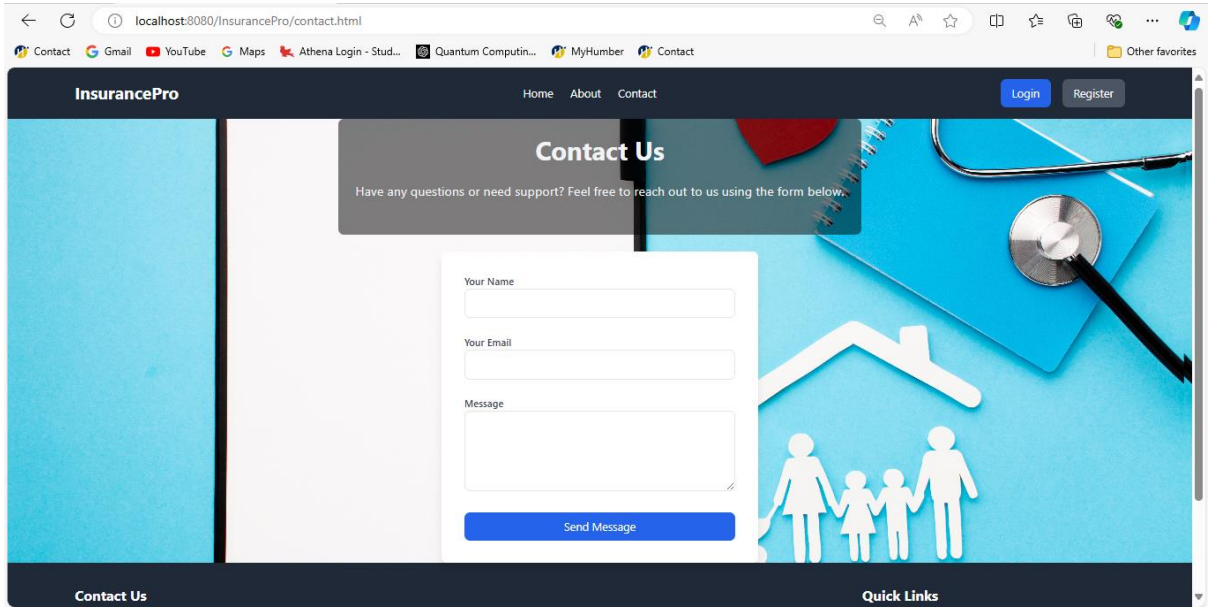
Home page



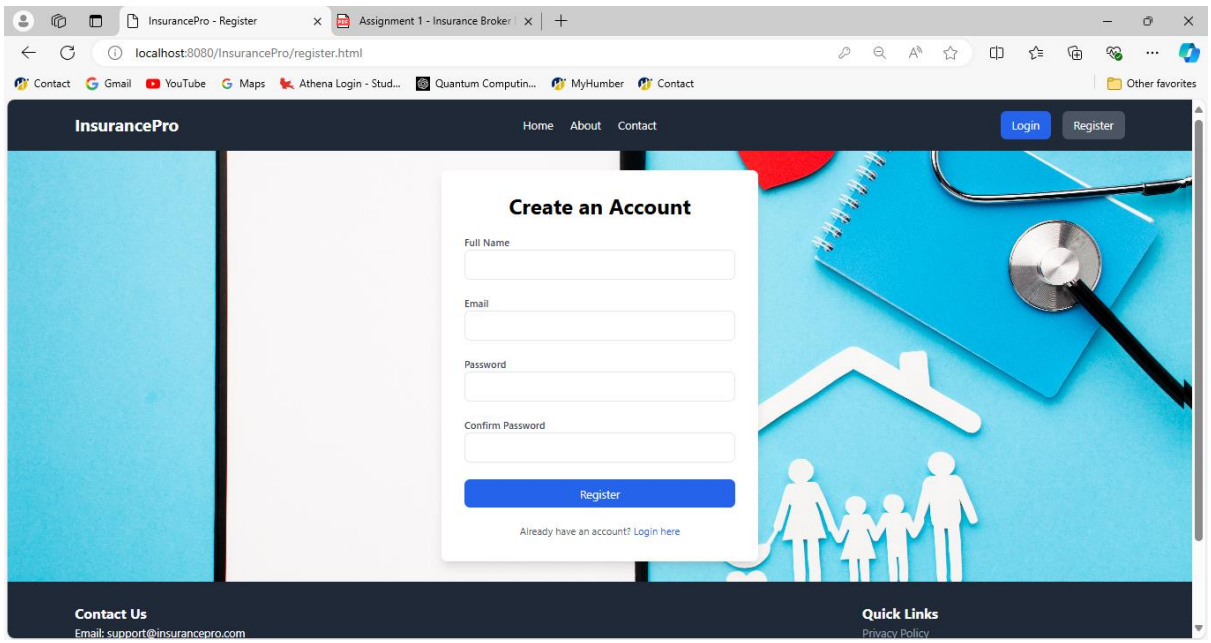
About page



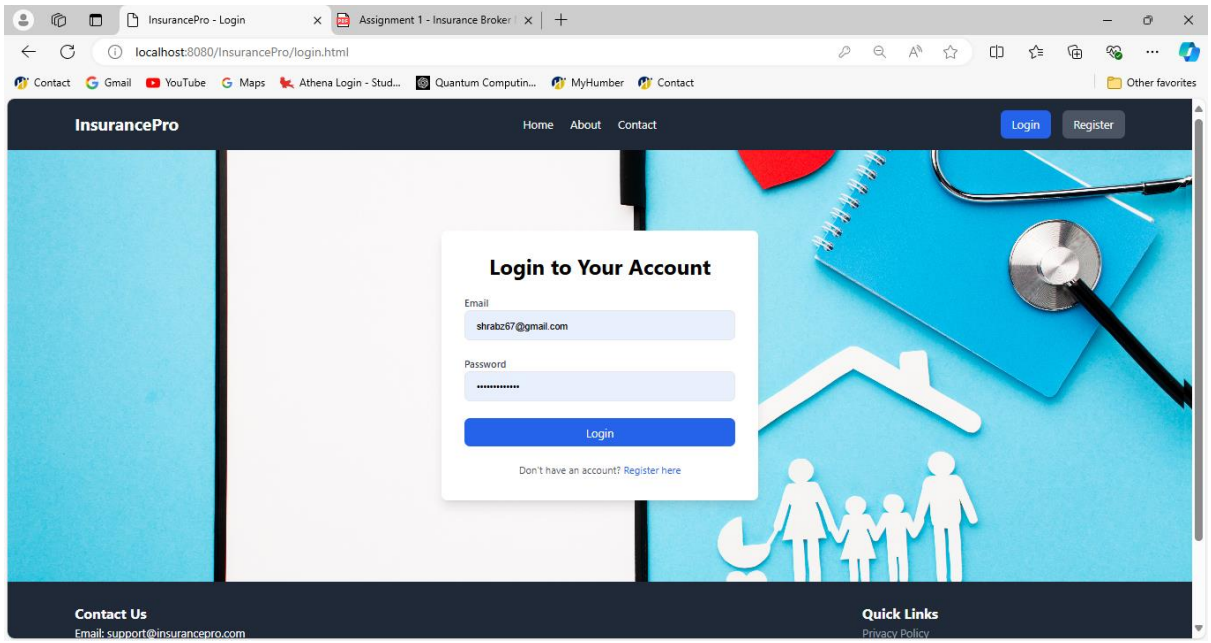
Contact Us page:-



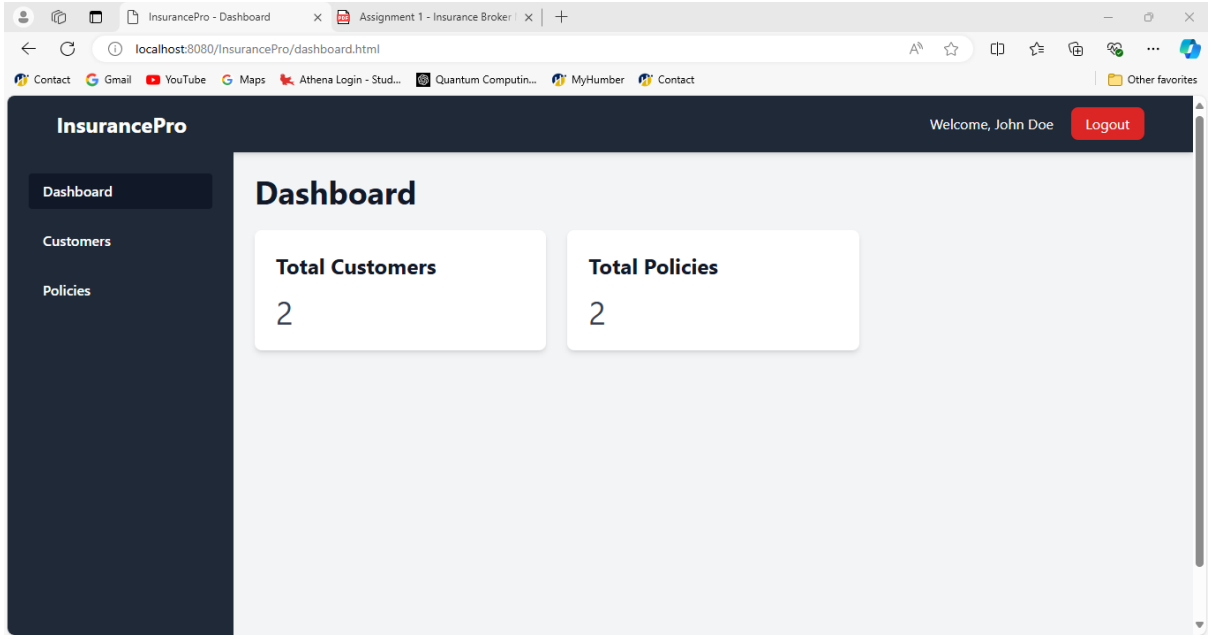
Register page



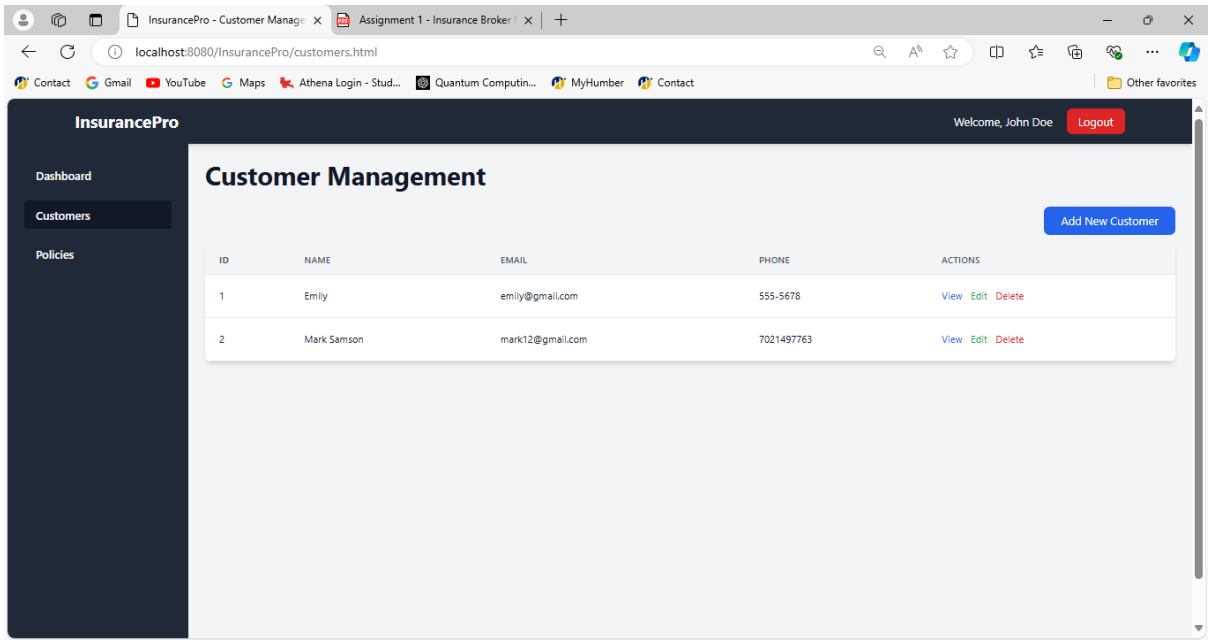
Login page



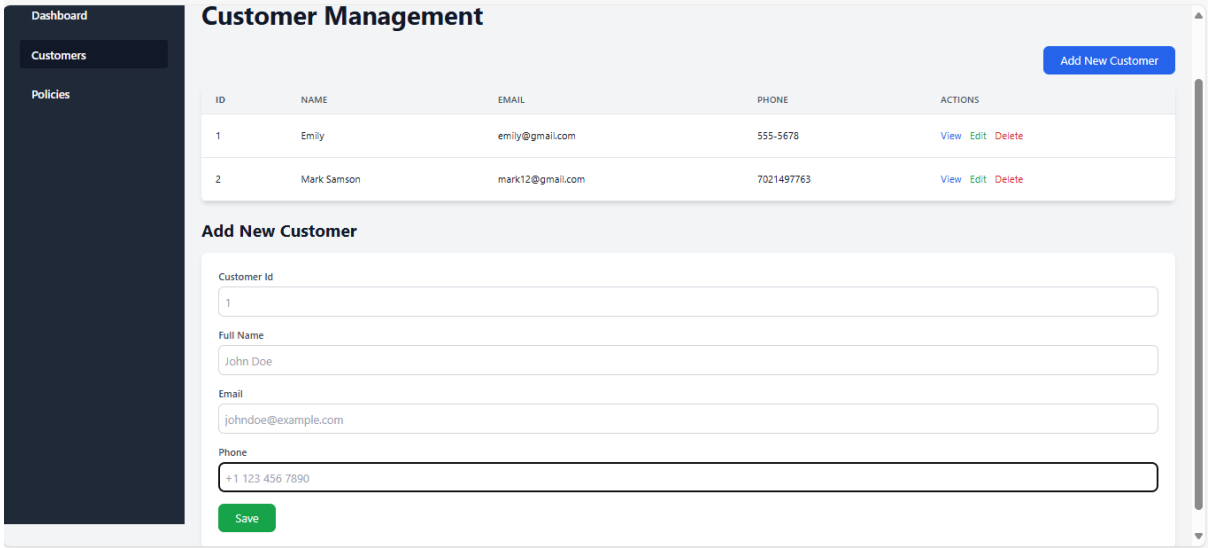
Dashboard page

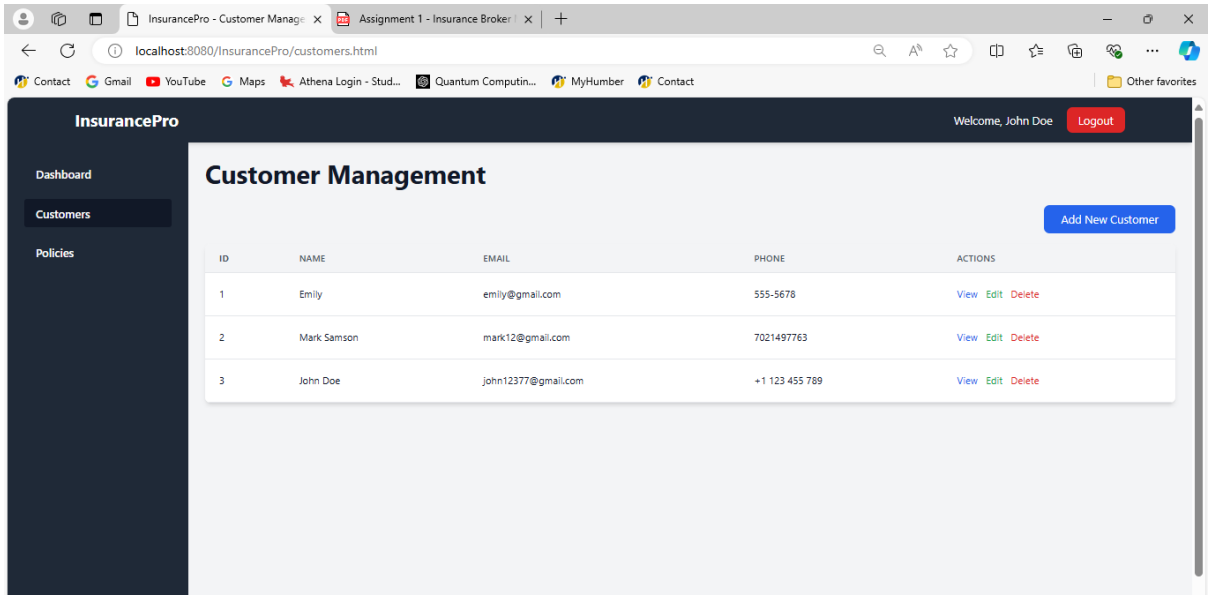


Customer Management page

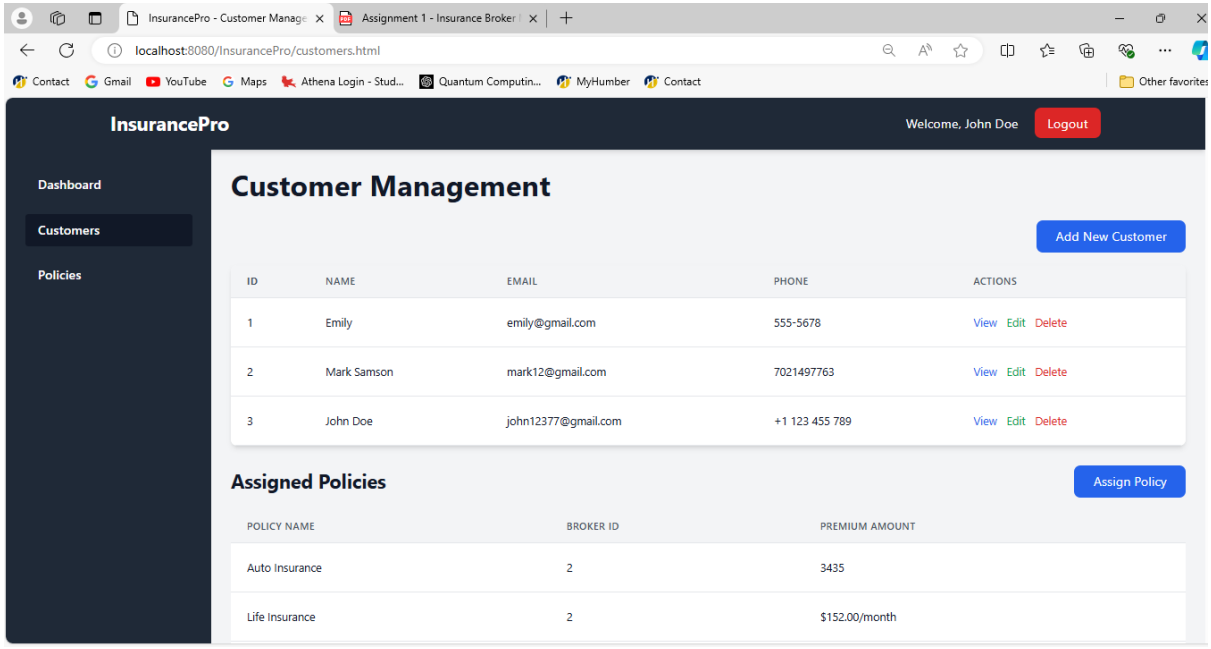


Add new customer functionality

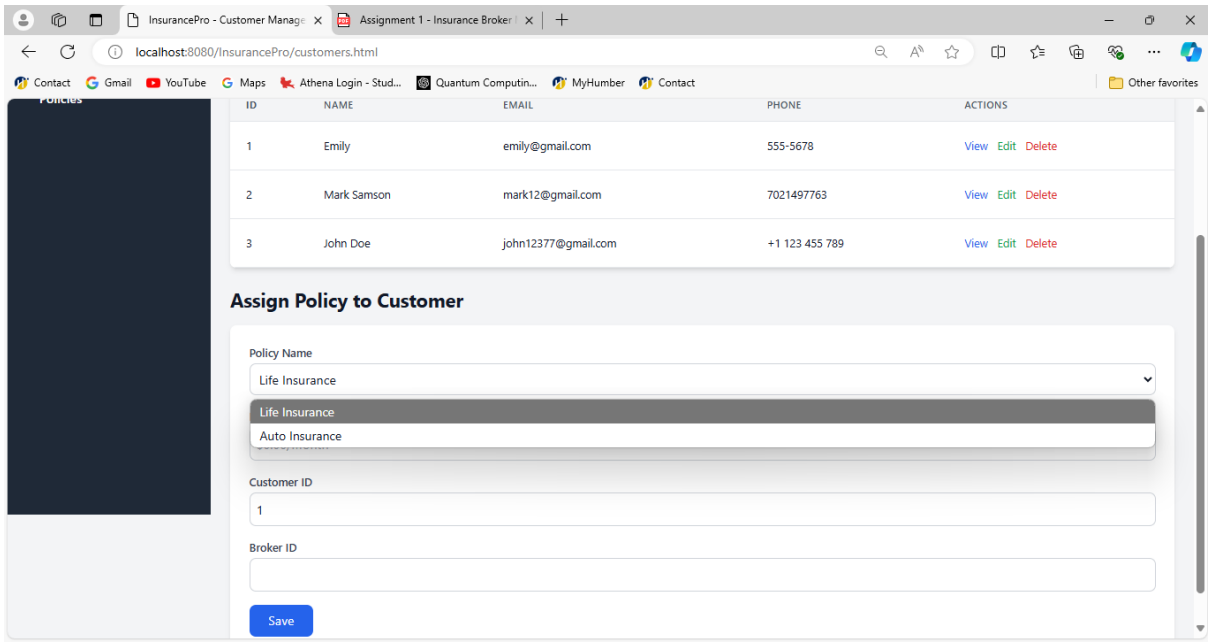




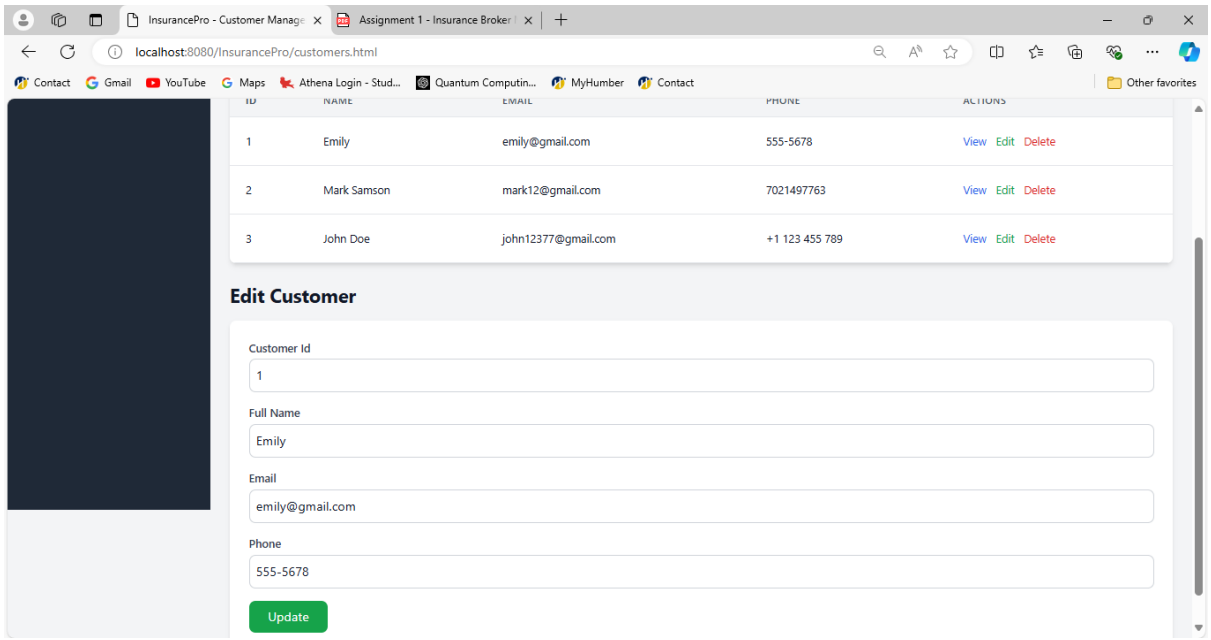
View functionality that displays the policy name ,broker Id and Premium Amount for the customer in Assigned policies table



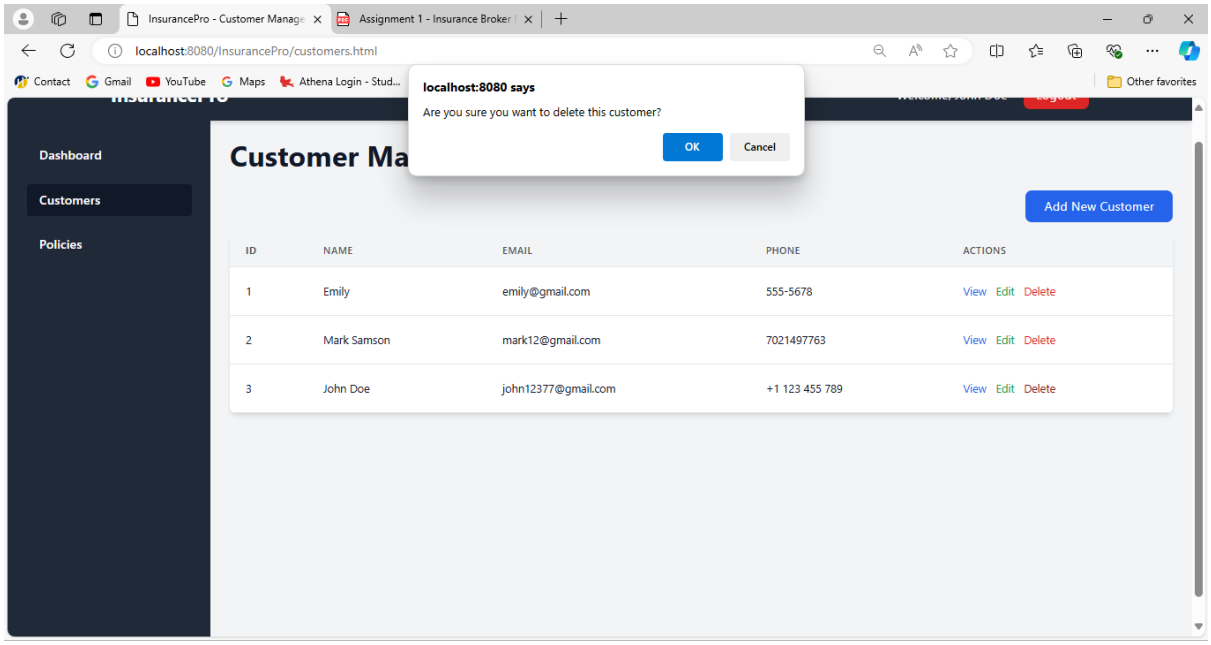
Assign Policy to customer functionality by clicking on Assign Policy button

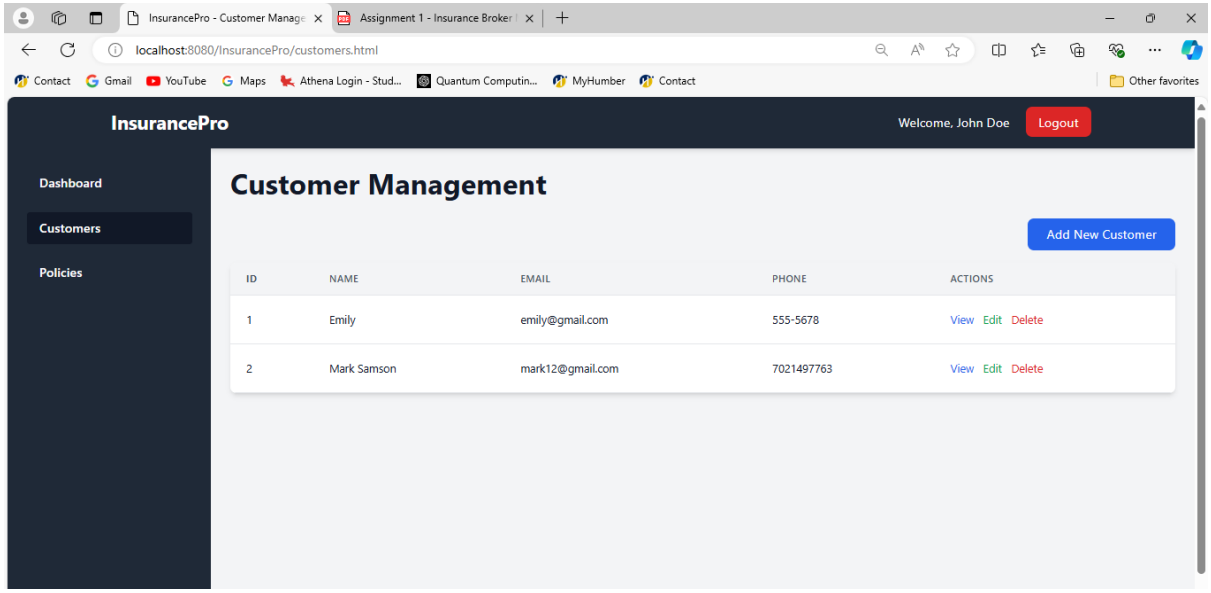


Edit functionality for customer

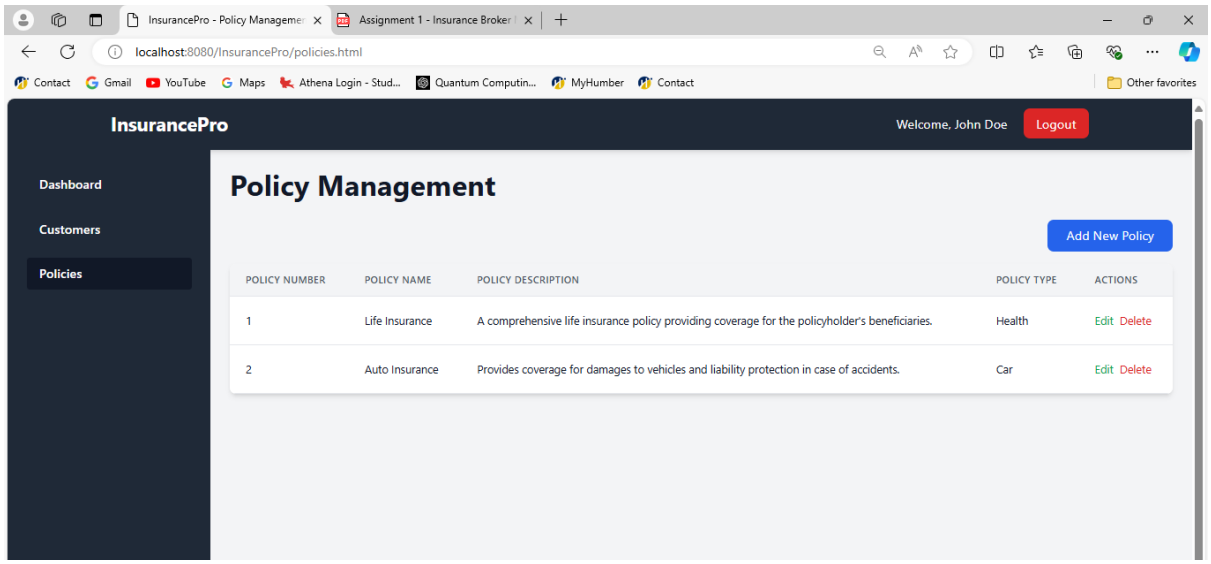


Delete functionality for customer

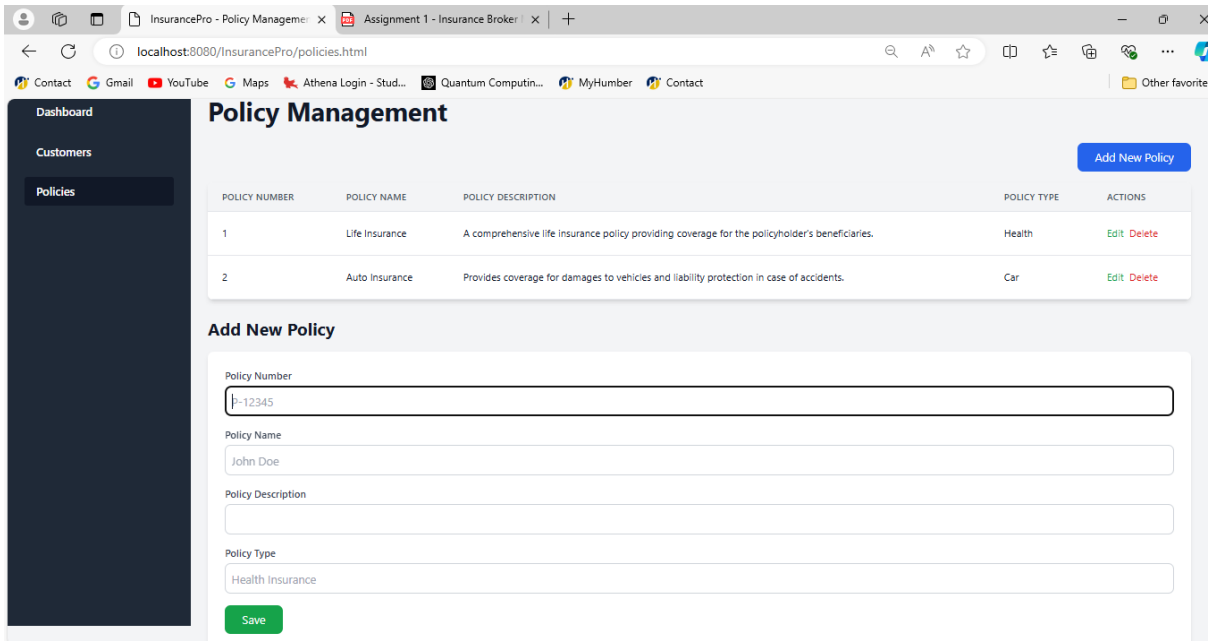


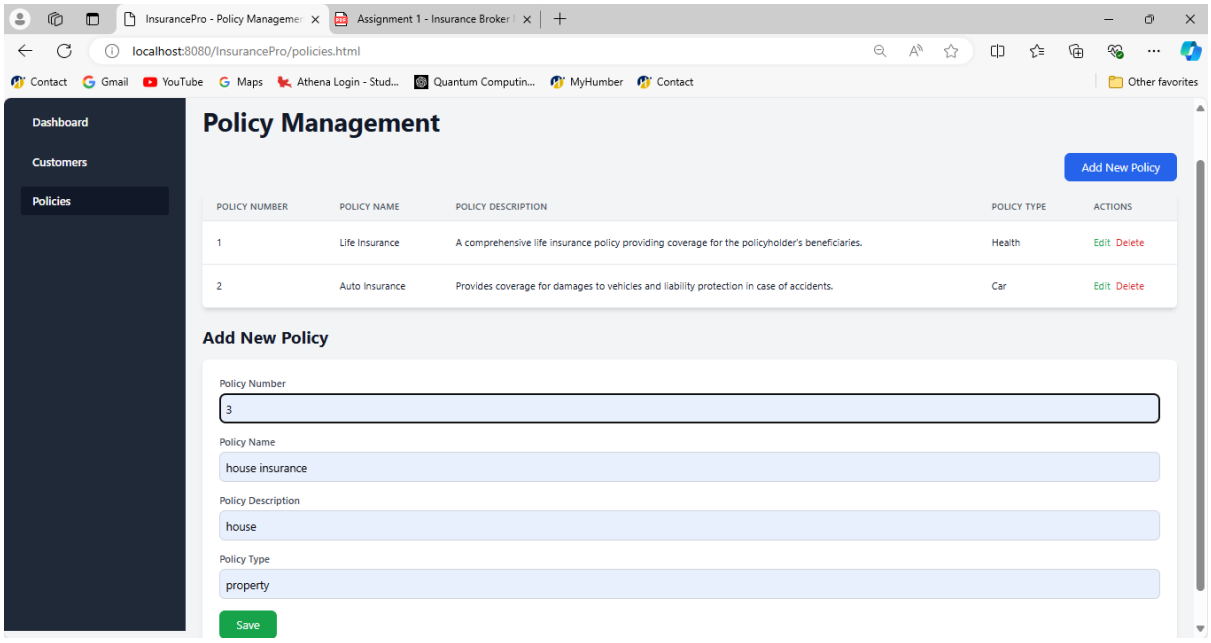


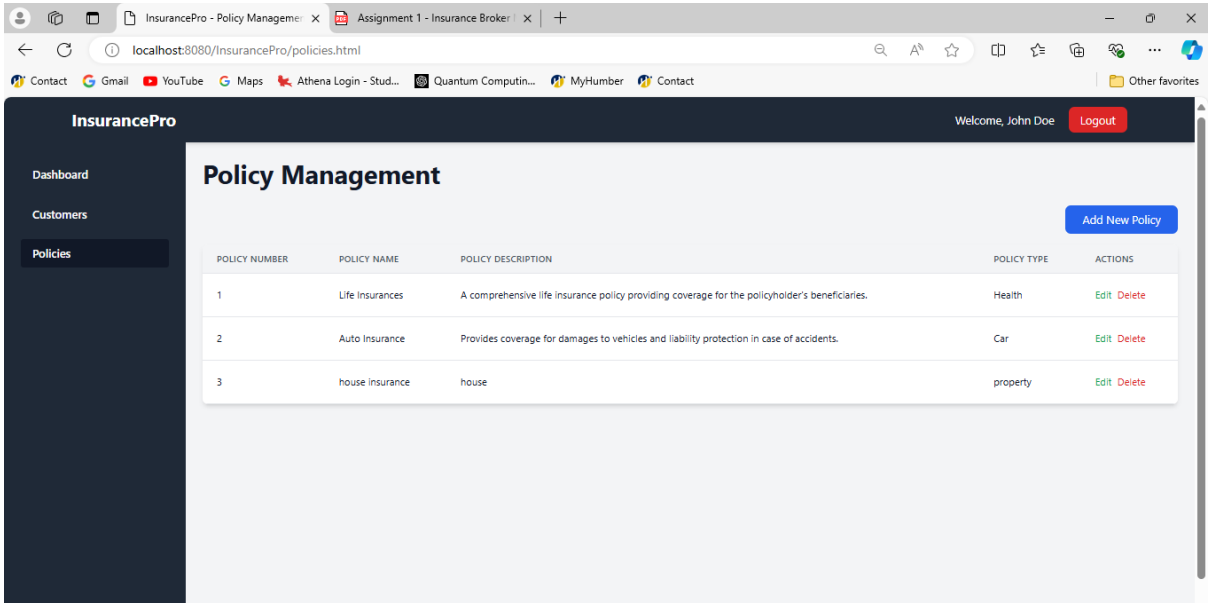
Policy Management page



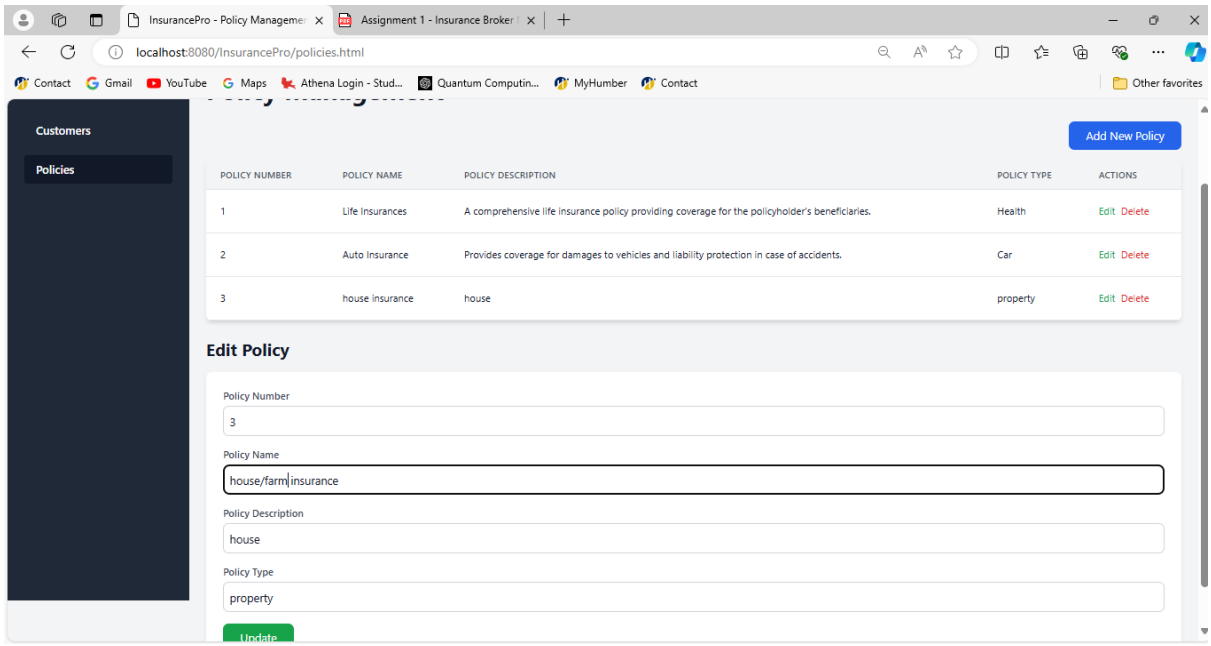
Add new policy functionality

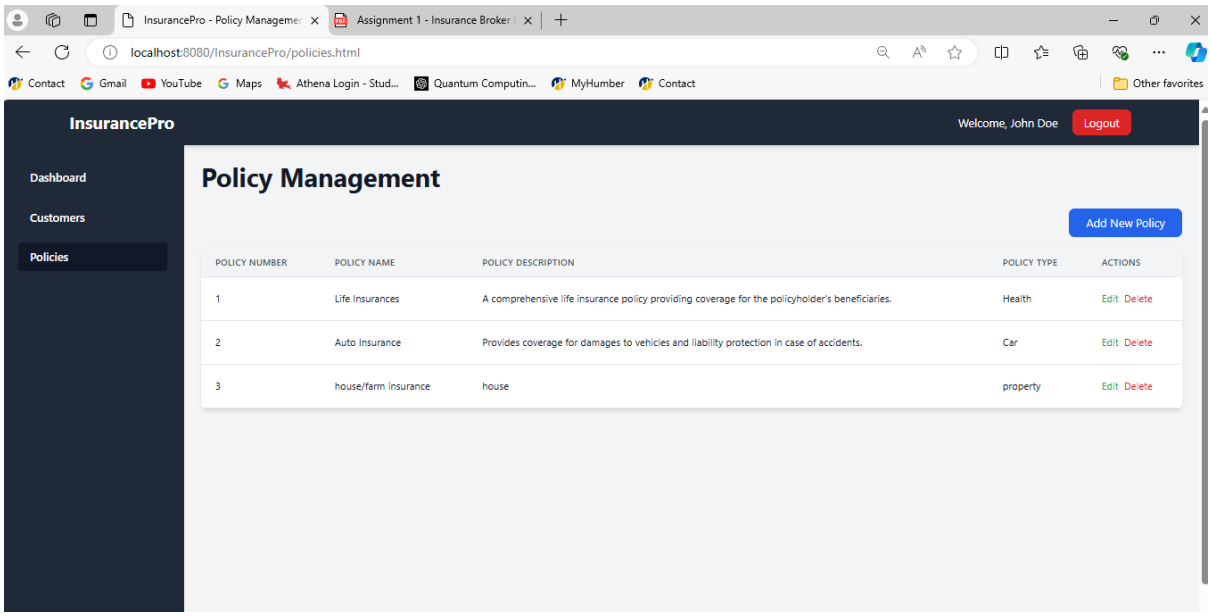



InsurancePro - Policy Manageme
Assignment 1 - Insurance Broker
localhost:8080/InsurancePro/policies.html
Contact
Gmail
YouTube
Maps
Athena Login - Stud...
Quantum Computin...
MyHumber
Contact
Other favorites
Dashboard
Customers
Policies
Policy Management
Add New Policy
POLICY NUMBER
POLICY NAME
POLICY DESCRIPTION
POLICY TYPE
ACTIONS
1
Life Insurance
A comprehensive life insurance policy providing coverage for the policyholder's beneficiaries.
Health
Edit Delete
2
Auto Insurance
Provides coverage for damages to vehicles and liability protection in case of accidents.
Car
Edit Delete
Add New Policy
Policy Number
3
Policy Name
house insurance
Policy Description
house
Policy Type
property
Save

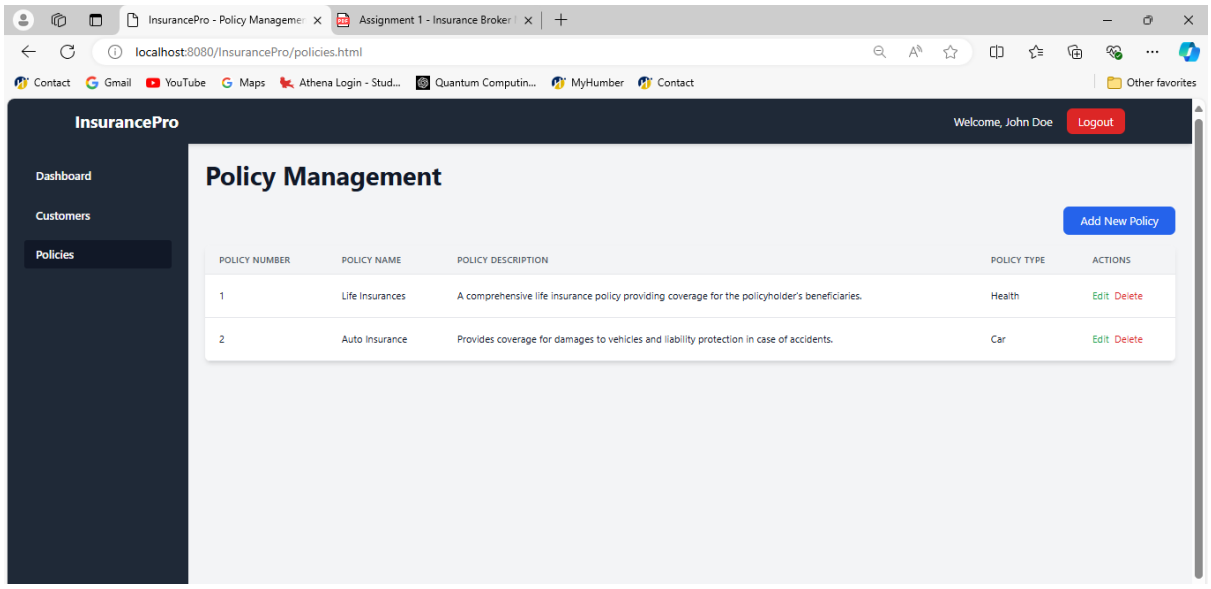

InsurancePro - Policy Manageme
Assignment 1 - Insurance Broker
localhost:8080/InsurancePro/policies.html
Contact
Gmail
YouTube
Maps
Athena Login - Stud...
Quantum Computin...
MyHumber
Contact
Other favorites
InsurancePro
Welcome, John Doe
Logout
Dashboard
Customers
Policies
Policy Management
Add New Policy
POLICY NUMBER
POLICY NAME
POLICY DESCRIPTION
POLICY TYPE
ACTIONS
1
Life Insurances
A comprehensive life insurance policy providing coverage for the policyholder's beneficiaries.
Health
Edit Delete
2
Auto Insurance
Provides coverage for damages to vehicles and liability protection in case of accidents.
Car
Edit Delete
3
house insurance
house
property
Edit Delete

Edit functionality





Delete functionality



A short video on concurrent access handling without data corruption displaying thread-safe resource management

[Double click to play]

20241005-1623-53.4
486973.mp4

Collaboration Details:

The team collaborated effectively using GitHub. All members contributed to different aspects of the project . Team members and the tasks assigned to each of the member:-

Back-end:-

1. CRUD operations for the Customer management : Samruddhi Chavan
2. Assign policies to customers : Samruddhi Chavan
3. CRUD operations for the policies management : Sruthi Jayaprakash Pandiath

Front-end:-

1. Home , About , Contact , Login, Register and Dashboard page :- Rajat Sachdeva
2. Add new customer, Add new policies and delete functionality in both pages :- Rajat Sachdeva
3. View , Assigned policies ,Assign policy to customer and Edit functionality in both pages:-Shrabani Sagareeka

UML Diagrams (Class and Sequence):- Manpreet Kaur Gulati

Report and Testing:- Shrabani Sagareeka

We used GitHub to track issues, assign tasks, and manage the project's progress.Conflicts were resolved by fetching the latest changes and resolving merge conflicts locally. The final version of the project was pushed to the repository after thorough testing.

Conclusion:

In conclusion, the Insurance Broker Management System effectively manages customer data and insurance policies using Java Servlets and JSON file storage. The system's architecture ensures separation of concerns and thread-safe operations. Team collaboration through GitHub facilitated smooth development, with each member contributing to key functionalities. The project meets the required objectives, providing a solid foundation for further development.