Zurich Assignment

## Architecture of the Quotation & Issuance

### Architecture:

Described the architecture of the Issuance and Quotation APIs in a visual format.

A diagram of a flowchart

Description automatically generated

### Description

* Implemented these API’s using “**Controller-Service-Repository**” Pattern:
  + **Controller**: It will Handle all HTTP requests and responses.
  + **Service**: Contain business logic.
  + **Repository**: Handle data persistence and retrieval.
* Each API is secured with Authentication
* In the Service Layer, a retry mechanism has been introduced, allowing each API to retry up to 3 times. If the API still fails after the 3rd attempt, an exception will be thrown.
* Additionally, a delay mechanism has been implemented in the Service Layer to accommodate core system execution delays.

### Database ER Diagrams

* Quotation ER Diagram:

A screenshot of a computer

Description automatically generated

* Issuance ER Diagram:

A screenshot of a computer

Description automatically generated

### Source code Management

* Committed the code to the Git repository, which can be accessed at: <https://github.com/phani238/Issuance-Quotation>
* I will attach the build file to the email for your reference.

## Concerned from Zurich for the Quotation & Issuance API

### Type of actions of the API

* **Quotation API**: Since the purpose of this API is to retrieve pricing information for an insurance product, the appropriate HTTP method used is GET.
* **Issuance API**: Since this API is used to issue a policy to the core system, the appropriate HTTP method used is POST.

### How to name the API

* **Quotation API**: /api/v1/quotations
* **Issuance API**: /api/v1/issuance

### How to version API when there is a breaking change?

* I have implemented URL versioning to manage API versions, as indicated in the JSON requests and responses:
  + <http://localhost:8888/api/v1/issuance>
  + <http://localhost:8888/api/v1/quotations>
* In the event of a breaking change, we can increment the version number and update the path accordingly (Ex: <http://localhost:8888/api/v2/issuance> , etc.)

### Design pattern of API

* Implemented using “**Controller-Service-Repository**” Pattern
  + **Controller**: It will Handle all HTTP requests and responses.
  + **Service**: Contain business logic.
  + **Repository**: Handle data persistence and retrieval.

### How should the JSON request and response look like? (Can please describe it in Open API Spec)

* Please find the attached JSON file, which contains all the requests and responses generated using Open API.



### How the API should react when there is a validation error or system error?

* Utilized appropriate HTTP status codes for error handling:
  + 400 Bad Request: For validation errors.
  + 500 Internal Server Error: For system errors.

### What happens when the core system is offline?

* Implemented a retry mechanism with exponential backoff for all Issuance and Quotation APIs.
* Each API will be retried up to 3 times. If the API still fails after the 3rd attempt, an exception will be thrown.
* For the Issuance API, a 2-minute delay has been implemented before execution, as the core system requires 2 minutes to process.