Insurance Policy Administration System

STAR life insurance company decided to automate its' policy administration system. Policy administrator should be able to define different type of insurance policies.

Each policy type will have the following information. Scheme number, Policy name (Ex. Jeevan Mitra, Jeevan Anurag, Jeevan Vishwas etc), Policy type (Endowment Term, Money back,), Maximum of number of years, Premium rate per year , Maximum sum assured (Ex. 10 Lakh, 20 Lakh etc.) 10, 10 lakh

Customer can take any type of policy available with STAR life insurance. Customer needs to provide the following personal information for registration. Customer name, age, sex, addresses and phone number.

Customer needs to specify the following to take a policy.

Policy name that he/she interested in, Sum Assured that he/she needs, No. of years that he/she wants to be insured, premium payment cycle (monthly/quarterly/annual) System validates that the sum assured and no of years given by the customer does not exceed the limits defined by Star. Also, the system generates the policy number for each policy. The system generates allocation id when customer chooses a policy and calculates the premium based on the premium rate defined in the scheme. And it assigns the premium to the policy based on the premium payment cycle. Customer can pay the amount online.

Requirements for Insurance Policy Administration Case study

1. Register Customer

The system has to register customers with their personal details. Once registration is successful, an id is issued to the customer.

2. Modify details

Admin can modify customer and policy details on receiving a request.

3. Update details

Admin can update the customer and policy details.

4. Add a New Policy

Admin can add a new policy to the database.

Generate Report

- View Customers based on customer id
- View Policy details based on policy id
- View customers allocated to a particular policy
- View customers based on policy status.
- Generate Bill after payment

Required Softwares

- Eclipse IDE
- Xampp / Mysql

Required Jar files

- mysql connector based on your mysql database version

Modules Required

The application should consist of 5 modules

- Customer Manipulation
- Policy Manipulation
- Allocate Policy to Customer
- Payment
- Claim Manipulation

Tables Required

Customer

Column Name	Data Type
CUSTOMER_ID	VARCHAR(25) **primary key**
CUSTOMER_NAME	VARCHAR(25)
DOB	DATE
AGE	INT
GENDER	VARCHAR(10)
OCCUPATION	VARCHAR(30)
ANNUAL_INCOME	DOUBLE
MEDICAL_HISTORY	VARCHAR(40)
ADDRESS	VARCHAR(255)
PHONE_NUMBER	BIGINT(20)
EMAIL_ID	VARCHAR(25)

Policy

Column Name	Data Type
POLICY_ID	VARCHAR(25) **primary key**
SCHEME_NUMBER	INT
POLICY_NAME	VARCHAR(50)
POLICY_TYPE	VARCHAR(30)
MAX_NO_OF_YEARS	INT
PREMIUM_RATE	DOUBLE
MAX_SUM_ASSURED	INT

Allocation

Column Name	Data Type
ALLOCATION_ID	VARCHAR(30) **primary key**

CUSTOMER_ID	VARCHAR(20) **foreign key**
POLICY_ID	VARCHAR(20) **foreign key**
NOMINEE_NAME	VARCHAR(30)
SUM_ASSURED	DECIMAL(10, 2)
NO_OF_YEARS	INT
PREMIUM_AMOUNT	DECIMAL(10, 2)
PREMIUM_PAYMENT_CYCLE	VARCHAR(20)
TOTAL_PAYMENT	DECIMAL(10, 2)
POLICY_STATUS	VARCHAR(20)

Payment

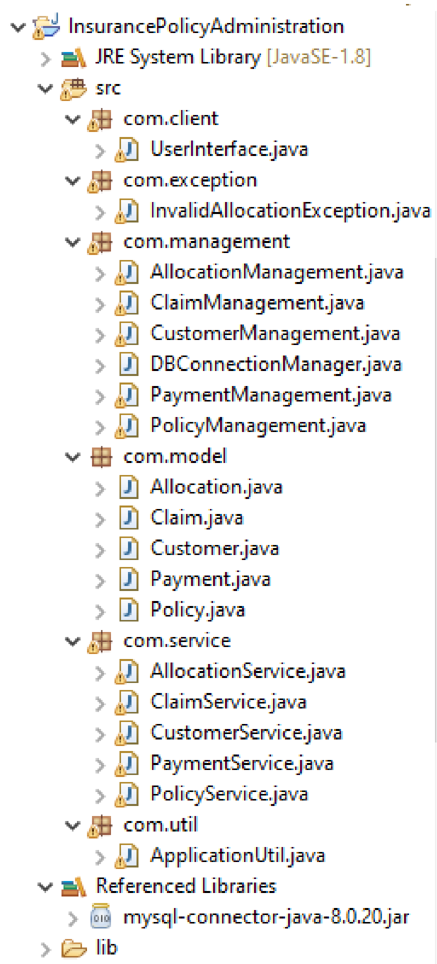
Column Name	Data Type
PAYMENT_ID	VARCHAR(30) **primary key**
CUSTOMER_ID	VARCHAR(30) **foreign key**
ALLOCATION_ID	VARCHAR(30) **foreign key**
PREMIUM	DOUBLE
PAYMENT_DATE	DATE
PAYMENT_MODE	VARCHAR(30)
INSTALLMENT_COUNT	INT

Claim

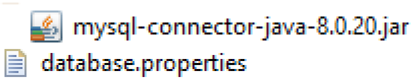
Column Name	Data Type
CLAIM_ID	VARCHAR(30) **primary key**
CUSTOMER_ID	VARCHAR(30) **foreign key**
ALLOCATION_ID	VARCHAR(30) **foreign key**
CLAIM_TYPE	VARCHAR(30)

CLAIM_AMOUNT	DOUBLE
CLAIM_DATE	DATE

Project Structure

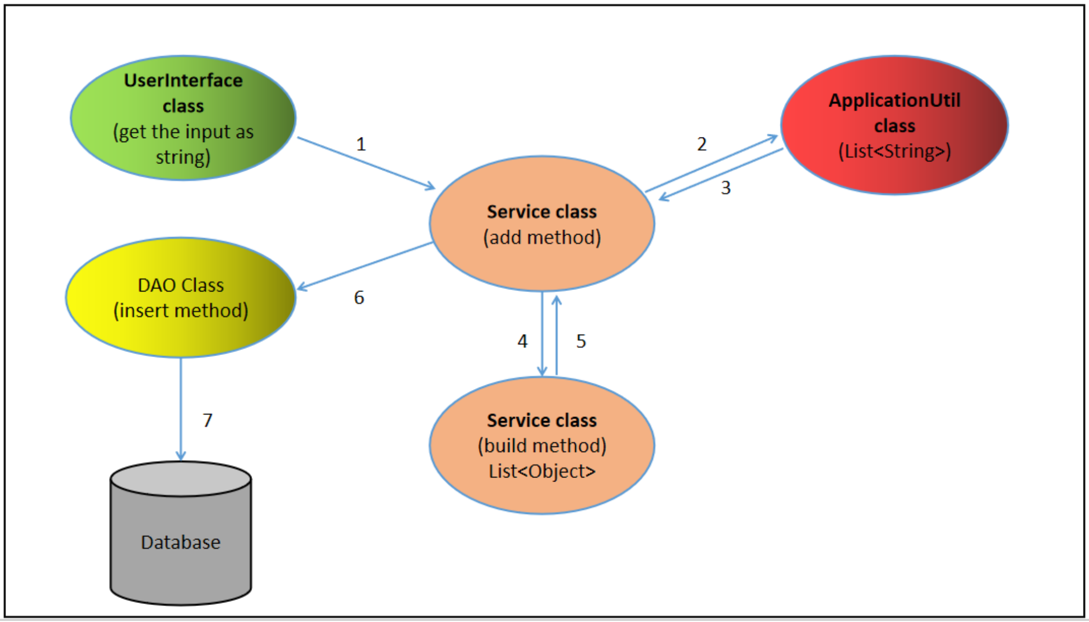


```
InsurancePolicyAdministration
  JRE System Library [JavaSE-1.8]
  src
    com.client
      UserInterface.java
    com.exception
      InvalidAllocationException.java
    com.management
      AllocationManagement.java
      ClaimManagement.java
      CustomerManagement.java
      DBConnectionManager.java
      PaymentManagement.java
      PolicyManagement.java
    com.model
      Allocation.java
      Claim.java
      Customer.java
      Payment.java
      Policy.java
    com.service
      AllocationService.java
      ClaimService.java
      CustomerService.java
      PaymentService.java
      PolicyService.java
    com.util
      ApplicationUtil.java
  Referenced Libraries
    mysql-connector-java-8.0.20.jar
  lib
```



```
    mysql-connector-java-8.0.20.jar
  database.properties
```

Project Flow:


UserInterface class (get the input as string)
1
Service class (add method)
2
3
ApplicationUtil class (List<String>)
DAO Class (insert method)
6
4
5
Service class (build method) List<Object>
7
Database

Scope of the Service module:

Module	Responsibility
CustomerService	The responsibility of the "**CustomerService**" module is to provide a set of services and functions related to customer management within an application. Its primary role is to act as an intermediary between the user interface and the data management module, typically referred to as "**CustomerManagement**." Here are the key responsibilities of the "**CustomerService**" module: **Customer Record Creation**: Provides functionality to create new customer records based on user inputs. Generates unique customer IDs for new customers to maintain data integrity. Ensures the validation and correct formatting of user inputs when creating customer records. **Customer Record Modification**: Offers services to modify customer information, such as updating phone numbers and email addresses. Ensures that data updates are correctly applied to the underlying data storage, typically a database. **Customer Data Retrieval**: Retrieves customer information based on customer IDs, allowing users to view and interact with customer records.

	Presents retrieved data to the user interface for display and potential modification. **Data Access**: Interfaces with the data management module, often referred to as "CustomerManagement," to execute operations like inserting, updating, and retrieving customer records in the database. Ensures that data operations are executed smoothly, maintaining data accuracy and consistency. **ID Generation**: Generates unique customer IDs for new customers based on a specific format, ensuring that the generated IDs are unique to prevent conflicts and maintain data consistency.
PolicyService	The responsibility of the "**PolicyService**" module is to provide a set of services and functions related to policy management within an application. It serves as an intermediary between the user interface and the data management module, typically referred to as "**PolicyManagement**." Here are the key responsibilities of the "**PolicyService**" module: **Policy Record Creation**: Provides functionality to create new policy records based on user inputs. Generates unique policy IDs for new policies to maintain data integrity. Ensures the validation and correct formatting of user inputs when creating policy records. **Policy Record Modification**: Offers services to modify policy information, such as updating maximum sum assured, maximum number of years, and other attributes. Ensures that data updates are correctly applied to the underlying data storage, typically a database. **Policy Data Retrieval**: Retrieves policy information based on policy IDs, allowing users to view and interact with policy records. Presents retrieved data to the user interface for display and potential modification. **Information Retrieval**: Provides methods to retrieve policy information, including the maximum sum assured, maximum number of years, and other policy attributes. Ensures that data is presented in a user-friendly format for display to users. **ID Generation**:

	Generates unique policy IDs for new policies based on a specific format, ensuring that the generated IDs are unique to prevent conflicts and maintain data consistency. **Data Deletion**: Provides services for deleting policy records based on policy IDs. Ensures the correct removal of policy data while handling exceptions that may occur during deletion.
AllocationService	The responsibility of the "**AllocationService**" module is to provide services and functions related to the allocation of policies within an insurance or financial application. It serves as an intermediary between the user interface and the data management modules, primarily "**AllocationManagement**" and "**PolicyManagement**." Here are the key responsibilities of the "**AllocationService**" module: **Allocation Creation**: Provides functionality to create new allocation records based on user inputs. Generates unique allocation IDs for new allocations to maintain data integrity. Calculates premium amounts for policies based on user-provided information. **Allocation Modification**: Offers services to modify allocation information, such as updating policy details and premium amounts. Ensures that data updates are correctly applied to the underlying data storage, typically a database. **Data Retrieval**: Retrieves allocation information based on various criteria, such as policy IDs and policy status. Presents retrieved data to the user interface for display and potential modification. **Policy Details Retrieval**: Retrieves policy details, such as premium rate, maximum sum assured, and maximum number of years, from the "PolicyManagement" module. Utilizes policy details to calculate premium amounts for allocations. **Data Conversion**: Utilizes utility methods, such as those provided by the "ApplicationUtil" class, for data conversion and validation. Ensures that data is correctly formatted for storage in the database and retrieval for display. **User Interaction**: Serves as the bridge between the user interface and the data management modules, enabling users to add new allocation records and modify existing records.

	Provides a user-friendly interface for managing allocation data. **Allocation Data Retrieval**: Provides methods to retrieve allocation information based on policy status and policy details. Ensures that data is presented in a user-friendly format for display to users. **ID Generation**: Generates unique allocation IDs for new allocations based on a specific format to prevent conflicts and maintain data consistency. **Data Deletion**: Provides services for deleting allocation records based on allocation IDs. Ensures the correct removal of allocation data while handling exceptions that may occur during deletion.
ClaimService	The "**ClaimService**" module is responsible for managing claims related to insurance policies. Its primary purpose is to handle claims submitted by customers and calculate claim amounts based on various claim types and policy details. Here are the key responsibilities of the "**ClaimService**" module: **Claim Creation**: Provides functionality to create new claim records based on user inputs. Generates unique claim IDs for new claims to maintain data integrity. Calculates claim amounts based on the claim type and associated allocation details. **Claim Calculation**: Calculates the claim amount based on the claim type (e.g., Death, Maturity, Survival benefit) and the corresponding allocation details (e.g., sum assured, number of years). Adjusts policy status based on the claim type, such as marking the policy as inactive for Death and Maturity claims. **Data Retrieval**: Retrieves claim information based on various criteria, such as customer IDs and claim types. Presents retrieved data to the user interface for display and potential modification. **Claim Amount Calculation**: Calculates the claim amount based on the claim type, taking into account the specific logic for different claim types. **Policy Details Retrieval**: Retrieves allocation details (e.g., sum assured, number of years) for claim amount calculation. Ensures that claim amounts are calculated accurately based on policy details.

	Claim Data Retrieval: Provides methods to retrieve claim information based on customer IDs and claim types. Ensures that claim data is presented in a user-friendly format for display to users. **ID Generation**: Generates unique claim IDs for new claims based on a specific format to prevent conflicts and maintain data consistency. **Data Deletion**: Provides services for deleting claim records based on claim IDs. Ensures the correct removal of claim data while handling exceptions that may occur during deletion.
PaymentService	The "**PaymentService**" module is responsible for managing payments within an insurance system. Its primary role is to handle payments made by customers for their insurance policies. Here are the key responsibilities of the "**PaymentService**" module: **Payment Creation**: Provides functionality to create new payment records based on user inputs. Generates unique payment IDs for new payments to maintain data integrity. Calculates the amount to be paid and records payment details, including the payment date, payment mode, and installment count. **Installment Count Calculation**: Calculates the installment count for a payment, taking into account the total payment made for an allocation. The installment count represents how many installments or payments have been made for a particular allocation. **Data Retrieval**: Retrieves payment information based on various criteria, such as payment IDs. Presents retrieved data in a user-friendly format for display and potential modification. **Payment Receipt Generation**: Generates payment receipts for customers who have made payments. The payment receipt includes details of the payment, confirmation messages, a payment reference number, contact information, and insurance company information. Provides a printable or digital payment receipt to customers for their records. **Payment Amount Calculation**: Calculates the payment amount for each installment, typically based on the premium amount of the associated insurance allocation.

	Ensures that payment amounts are accurately determined according to the policy details. **Data Update**: Updates allocation details, specifically the total payment for an allocation, once a payment is recorded. Ensures that allocation data reflects the total payments made by customers. **ID Generation**: Generates unique payment IDs for new payments based on a specific format to prevent conflicts and maintain data consistency. **Data Deletion**: Provides services for deleting payment records based on payment IDs. Ensures the correct removal of payment data while handling exceptions that may occur during deletion.

Scope of the management modules:

Module	Responsibility
CustomerManagement	The "CustomerManagement" module is responsible for managing customer data within an insurance system. It interacts with the database to perform various operations related to customer information. Here are the key responsibilities of the "CustomerManagement" module: **Customer Data Insertion:** Inserts new customer records into the database. This operation is typically used to add customer information to the system. **Customer Data Update (Phone Number):** Updates the phone number of a customer based on their customer ID. This allows customers to change or update their contact information. **Customer Data Update (Email ID):** Updates the email address of a customer based on their customer ID. This operation allows customers to change their email address, which is essential for communication. **ID Existence Check:** Checks if a given customer ID already exists in the database. This is useful for ensuring the uniqueness of customer IDs. **Customer Data Retrieval:** Retrieves customer data based on customer ID.

	Retrieves all customer information related to a specific customer from the database. This information can be used for various purposes within the system. **Customer Data Deletion:** Deletes a customer record based on customer ID. This operation is used when a customer wants to remove their information from the system.
PolicyManagement	The "**PolicyManagement**" module is responsible for managing policy data within an insurance system. It interacts with the database to perform various operations related to policy information. Here are the key responsibilities of the "**PolicyManagement**" module: **Policy Data Insertion:** Inserts new policy records into the database. This operation is typically used to add new insurance policies to the system. **Policy Data Retrieval (Based on Policy ID):** Retrieves policy data based on the policy ID. This operation allows the system to fetch details of a specific policy. **ID Existence Check:** Checks if a given policy ID already exists in the database. This is useful for ensuring the uniqueness of policy IDs. **Policy Data Update (Maximum Sum Assured):** Updates the maximum sum assured value for a policy based on its policy ID. This operation is used to modify policy parameters. **Policy Data Update (Maximum Number of Years):** Updates the maximum number of years for a policy based on its policy ID. This operation is used to modify policy parameters. **Policy Data Deletion:** Deletes a policy record based on policy ID. This operation is used when a policy is discontinued or removed from the system.
AllocationManagement	The "AllocationManagement" module is responsible for managing allocation data within an insurance system. It interacts with the database to perform various operations related to allocation information. Here are the key responsibilities of the "AllocationManagement" module: **Allocation Data Insertion:** Inserts new allocation records into the database. This operation is typically used to create new insurance policies for customers. **Allocation Data Update (Total Payment):** Updates the total payment value for an allocation based on its allocation ID. This operation is used to keep track of the total amount paid by customers. **Allocation Data Update (Policy Status):**

	Updates the policy status for an allocation based on its allocation ID. This operation is used to mark policies as "closed" when a customer claims or the policy matures. **Policy Retrieval for Allocation:** Retrieves policy information associated with an allocation based on the policy ID. This is used to obtain policy details for a specific allocation. **Allocation Data Retrieval (Based on Policy ID):** Retrieves allocation data based on the policy ID. This operation allows the system to fetch all allocations associated with a specific policy. **Allocation Data Retrieval (Based on Policy Status):** Retrieves allocation data based on the policy status. This is useful for finding allocations with a specific policy status, such as "closed." **Allocation Data Retrieval (Based on Allocation ID):** Retrieves allocation data based on the allocation ID. This operation is useful for obtaining details of a specific allocation.
PaymentManagement	The "**PaymentManagement**" module is responsible for managing payment data within an insurance system. It interacts with the database to perform various operations related to payment information. Here are the key responsibilities of the "PaymentManagement" module: **Payment Data Insertion:** Inserts new payment records into the database. This operation is used to record customer payments for insurance policies. **Payment Data Retrieval:** Retrieves payment information based on the payment ID. This operation allows the system to fetch details of a specific payment.
ClaimManagement	The "**ClaimManagement**" module is responsible for managing claim-related data within an insurance system. It interacts with the database to perform various operations related to claim information. Here are the key responsibilities of the "**ClaimManagement**" module: **Claim Data Insertion**: Inserts new claim records into the database. This operation is used to record customer claims for insurance policies. **Claim Data Retrieval**: Retrieves claim information based on customer ID. This operation allows the system to fetch details of claims associated with a specific customer.

Scope of Database module:

Module	Responsibilities
DBConnectionManager	**The DBConnectionManager module** is responsible for managing and providing database connections to other modules in the application. Its primary responsibilities include: **Loading Database Properties:** It reads the database connection properties from a configuration file, such as database.properties, which typically contains details like the database URL, driver name, username, and password. **Establishing Database Connection:** It establishes a connection to the database using the properties loaded from the configuration file. This connection is created through the Java Database Connectivity (JDBC) API. **Handling Exceptions:** It handles exceptions related to file IO (e.g., IOException), database drivers (e.g., ClassNotFoundException), and SQL errors (e.g., SQLException) that may occur during the connection process.

Scope of the Exception module

Module Name	Responsibility
InvalidAllocationException	The **InvalidAllocationException** class is a custom exception class used to handle exceptions related to invalid allocation records in the application. It extends the standard Exception class and includes a single-argument constructor to set the exception message. The exception message is passed to the super class constructor.