|  |  |
| --- | --- |
|  |  |
| Policy Project |  |
|  |  |
|  | 9th OctoberGroup 5 |
|  | Anuj SharmaPreet Bobde |
|  | Chakrala Sai MadhavLavkush Kumar |
|  | Veerain Sood |

Class Control0:

CreateCustomer(): This method allows the user to input customer information like name, phone number, address, and salary. It generates a unique user ID and a random PIN for the customer. It also ensures that the phone number has exactly 10 digits and that the salary is above a certain threshold.

Interface(): This method is used for customer login. It prompts the user to enter their user ID and PIN and then calls the CusInterface() method to provide various options for customers, such as viewing details, getting a policy, or managing their insurance.

CusInterface(): This method handles customer interactions after login. It offers a menu of options to customers, including viewing details, getting a policy, upgrading premium, claiming insurance, and viewing claim status. Depending on the user's choice, it calls various methods from the Agent class to perform these actions.

iaInterface(): This method appears to be intended for interactions with an insurance agent (IA). It presents options for an insurance agent to view customer data and approve insurance claims. The agent can view all customers, customers with pending home, life, or car insurance claims, and approve these claims.

ApproveClaim(): This method is used to approve insurance claims for a specific customer. It prompts the agent to enter the user ID, shows customer details, and allows the agent to approve pending claims for home, life, or car insurance.

ViewCustomer(): This method displays customer data, including user ID, name, and the status of their insurance policies. It can show data for all customers or filter by pending claims for different types of insurance (home, life, car).

Class Customer :

Instance Variables:

name: Stores the customer's name.

address: Stores the customer's address.

phoneNum: Stores the customer's phone number.

salary: Stores the customer's salary.

userId: Stores the customer's unique user ID.

Password: Stores the customer's password.

incident: An array of strings to store information about incidents related to insurance policies.

premium: An integer flag (0 for normal, 1 for premium) indicating the customer's premium status.

status: An array of integers (size 3) representing the status of insurance policies (0 for inactive, 1 for active, 2 for pending).

Constructor:

The constructor initializes the customer's attributes such as name, address, phone number, user ID, password, and salary.

ViewDetails(int f):

This method displays customer details, including user ID, password (only if f is 1) and other details.

The status array is used to display the status of insurance policies for the customer.

The incident array stores information about incidents related to insurance policies.

The premium status is also displayed.

Getter and Setter Methods: The class provides getter and setter methods for various attributes like name, address, phone number, salary, user ID, and password.

Policy:

PolicyUtil Class:

getIntInput(): Helper method to get an integer input from the user, handling exceptions and invalid input.

clearInputBuffer(): Helper method to clear the input buffer (not currently used).

Policy Abstract Class:

Defines common attributes and methods for all types of insurance policies.

AwardPolicy(Customer c): A method to determine if a policy can be awarded to a customer based on certain criteria, such as salary, smoking, disaster-prone area, etc. It returns a boolean indicating if the policy is granted.

PolicyClaimEligibility(Customer c): A method to check if a customer is eligible to claim a policy, considering their policy status and incident description.

homeInsurance Class (extends Policy):

Policy(Customer c):

Abstract class for home insurance , car insurence and automobile insurance

home insurance:

Implements specific criteria for granting home insurance policies.

Checks for conditions like salary, home ownership, and living in disaster-prone areas.

Overrides AwardPolicy(Customer c) and PolicyClaimEligibility(Customer c) methods.

Lifeinsurance Class (extends Policy):

life insurance policies:

Implements specific criteria for granting life insurance policies.

Checks for conditions like salary, smoking, drinking, cancer, and AIDS.

Overrides AwardPolicy(Customer c) and PolicyClaimEligibility(Customer c) methods.

automobileInsurance Class (extends Policy):

automobile insurance policies:

Implements specific criteria for granting automobile insurance policies.

Checks for conditions like salary, pollution check certificate, and legal driving licenses.

Overrides AwardPolicy(Customer c) and PolicyClaimEligibility(Customer c) methods.

Agent:

Policy(Customer c):

Presents a menu for customers to select the type of insurance policy they want (Life Insurance, Home Insurance, Car Insurance, or to go back).

Calls the respective policy's AwardPolicy(Customer c) method based on the customer's choice to determine if the policy can be granted.

createPolicy(Customer c):

Presents a menu for customers to select the type of insurance policy for which they want to claim (Life Insurance, Home Insurance, Car Insurance, or to go back).

Calls the respective policy's PolicyClaimEligibility(Customer c) method based on the customer's choice to check if they are eligible to claim the policy.

ViewCustomer(Customer c):

Displays customer information and the status of their insurance policies in a tabular format.

DataBase: This is static and stores all the information and customer objects

customers: This is an ArrayList that stores Customer objects, representing a database of customers.

CustomerIndex(String userId):

Takes a userId as input.

Iterates through the customers list to find a customer with a matching userId.

If a matching customer is found, it returns the index of that customer in the list.

If no matching customer is found, it returns -1 to indicate that the customer with the given userId does not exist in the database.

CheckPassword(int index, String Password):

Takes an index (obtained from CustomerIndex) and a Password as input.

Retrieves the Customer object at the specified index in the customers list.

Compares the provided Password with the password stored in the customer's object.

If the passwords match, it returns 1 to indicate a successful password match.

If the passwords do not match, it returns 0 to indicate that the password is incorrect.