JAVA FOUNDATION TRAINING

CODING CHALLENGE

Insurance – JAVA

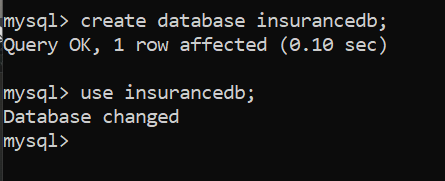
NAME: Sherin Sandra J

BATCH: 3

TRAINER: Mr. Madhu Kalla

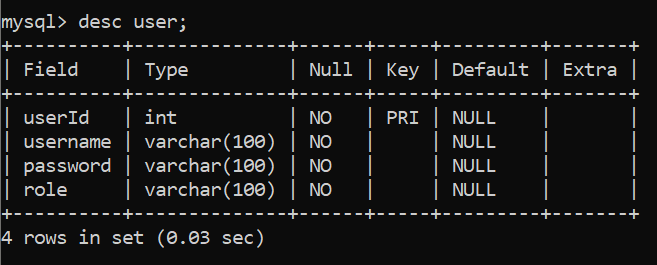
DATE: 09/04/2025

1. **Creating SQL Schema from the following classes class, use the class attributes for table column names.**



\* create table user( userId int primary key, username varchar(100) not null, password varchar(100) not null, role varchar(100) not null);

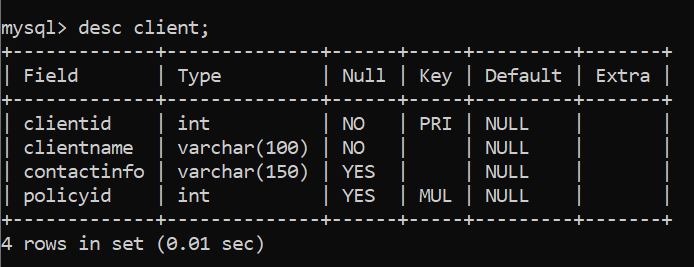
Query OK, 0 rows affected (0.23 sec)



\* create table client (

-> clientid int primary key, clientname varchar(100) not null, contactinfo varchar(150), policyid int, foreign key (policyid) references policy(policyid) );

Query OK, 0 rows affected (0.09 sec)

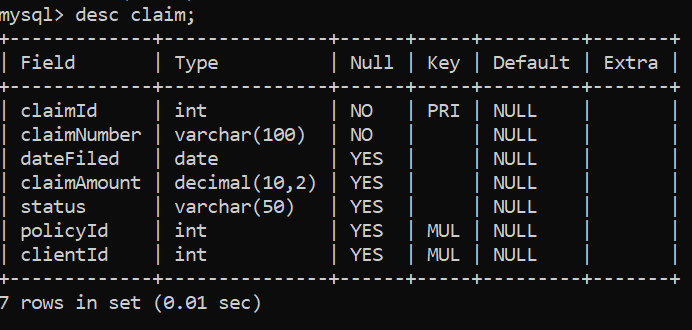


\* create table claim ( claimId int primary key, claimNumber varchar(100) not null, dateFiled date, claimAmount decimal(10,2), status varchar(50), policyId int, clientId int,

foreign key (policyId) references policy(policyId),

foreign key (clientId) references client(clientId) );

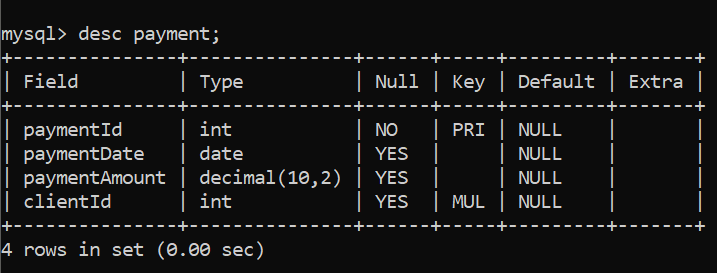
Query OK, 0 rows affected (0.08 sec)



\* create table payment (

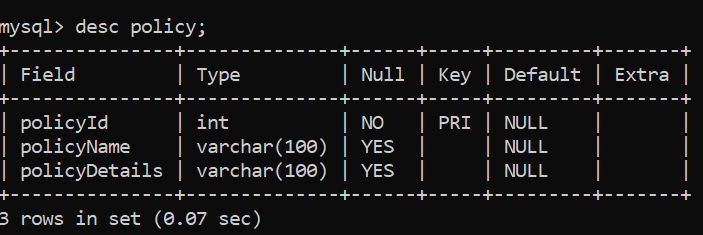
-> paymentId int primary key, paymentDate date, paymentAmount decimal(10,2), clientId int, foreign key (clientId) references client(clientId) );

Query OK, 0 rows affected (0.08 sec)



\* create table policy( policyId int primary key, policyName varchar(100), policyDetails varchar(100));

Query OK, 0 rows affected (0.05 sec)



1. **Java Package Structure**

| Package Name | Purpose |
| --- | --- |
| hexa.org.entity | Holds entity classes |
| hexa.org.dao | Contains DAO interfaces and implementation |
| hexa.org.exception | Custom exceptions (PolicyNotFoundException) |
| hexa.org.util | Utility classes for DB connection |
| hexa.org.main | MainModule.java |

1. **Creating the following model/entity classes within package entity with variables declared private, constructors(default and parametrized,getters,setters and toString())**

**Policy.java**

|  |
| --- |
| package hexa.org.entity;  public class Policy {    private int policyId;  private String policyName;  private String policyDetails;  public Policy() {  super();  }  public Policy(int policyId, String policyName, String policyDetails) {  super();  this.policyId = policyId;  this.policyName = policyName;  this.policyDetails = policyDetails;  }  public int getPolicyId() {  return policyId;  }  public void setPolicyId(int policyId) {  this.policyId = policyId;  }  public String getPolicyName() {  return policyName;  }  public void setPolicyName(String policyName) {  this.policyName = policyName;  }  public String getPolicyDetails() {  return policyDetails;  }  public void setPolicyDetails(String policyDetails) {  this.policyDetails = policyDetails;  }  @Override  public String toString() {  return "Policy [policyId=" + policyId + ", policyName=" + policyName + ", policyDetails=" + policyDetails + "]";  }  } |

**Client.java**

|  |
| --- |
| package hexa.org.entity;  public class Client {    private int clientId;  private String clientName;  private String contactInfo;  private Policy policy;  public Client() {  super();  }  public Client(int clientId, String clientName, String contactInfo, Policy policy) {  super();  this.clientId = clientId;  this.clientName = clientName;  this.contactInfo = contactInfo;  this.policy = policy;  }  public int getClientId() {  return clientId;  }  public void setClientId(int clientId) {  this.clientId = clientId;  }  public String getClientName() {  return clientName;  }  public void setClientName(String clientName) {  this.clientName = clientName;  }  public String getContactInfo() {  return contactInfo;  }  public void setContactInfo(String contactInfo) {  this.contactInfo = contactInfo;  }  public Policy getPolicy() {  return policy;  }  public void setPolicy(Policy policy) {  this.policy = policy;  }  @Override  public String toString() {  return "Client [clientId=" + clientId + ", clientName=" + clientName + ", contactInfo=" + contactInfo  + ", policy=" + policy + "]";  }} |

**User.java**

|  |
| --- |
| package hexa.org.entity;  public class User {  private int userId;  private String userName;  private String password;  private String role;  public User() {  super();  }  public User(int userId, String userName, String password, String role) {  super();  this.userId = userId;  this.userName = userName;  this.password = password;  this.role = role;  }  public int getUserId() {  return userId;  }  public void setUserId(int userId) {  this.userId = userId;  }  public String getUserName() {  return userName;  }  public void setUserName(String userName) {  this.userName = userName;  }  public String getPassword() {  return password;  }  public void setPassword(String password) {  this.password = password;  }  public String getRole() {  return role;  }  public void setRole(String role) {  this.role = role;  }  @Override  public String toString() {  return "User [userId=" + userId + ", userName=" + userName + ", password=" + password + ", role=" + role + "]";  }  } |

**Claim.java**

|  |
| --- |
| package hexa.org.entity;  import java.util.Date;  public class Claim {    private int claimId;  private String claimNumber;  private Date dateFiled;  private double claimAmount;  private String status;  private Policy policy;  private Client client;  public Claim() {  super();  }  public Claim(int claimId, String claimNumber, Date dateFiled, double claimAmount, String status, Policy policy,  Client client) {  super();  this.claimId = claimId;  this.claimNumber = claimNumber;  this.dateFiled = dateFiled;  this.claimAmount = claimAmount;  this.status = status;  this.policy = policy;  this.client = client;  }  public int getClaimId() {  return claimId;  }  public void setClaimId(int claimId) {  this.claimId = claimId;  }  public String getClaimNumber() {  return claimNumber;  }  public void setClaimNumber(String claimNumber) {  this.claimNumber = claimNumber;  }  public Date getDateFiled() {  return dateFiled;  }  public void setDateFiled(Date dateFiled) {  this.dateFiled = dateFiled;  }  public double getClaimAmount() {  return claimAmount;  }  public void setClaimAmount(double claimAmount) {  this.claimAmount = claimAmount;  }  public String getStatus() {  return status;  }  public void setStatus(String status) {  this.status = status;  }  public Policy getPolicy() {  return policy;  }  public void setPolicy(Policy policy) {  this.policy = policy;  }  public Client getClient() {  return client;  }  public void setClient(Client client) {  this.client = client;  }  @Override  public String toString() {  return "Claim [claimId=" + claimId + ", claimNumber=" + claimNumber + ", dateFiled=" + dateFiled  + ", claimAmount=" + claimAmount + ", status=" + status + ", policy=" + policy + ", client=" + client  + "]";  }  } |

**Payment.java**

|  |
| --- |
| package hexa.org.entity;  import java.util.Date;  public class Payment {  private int paymentId;  private Date paymentDate;  private double paymentAmount;  private Client client;  public Payment() {  super();  }  public Payment(int paymentId, Date paymentDate, double paymentAmount, Client client) {  super();  this.paymentId = paymentId;  this.paymentDate = paymentDate;  this.paymentAmount = paymentAmount;  this.client = client;  }  public int getPaymentId() {  return paymentId;  }  public void setPaymentId(int paymentId) {  this.paymentId = paymentId;  }  public Date getPaymentDate() {  return paymentDate;  }  public void setPaymentDate(Date paymentDate) {  this.paymentDate = paymentDate;  }  public double getPaymentAmount() {  return paymentAmount;  }  public void setPaymentAmount(double paymentAmount) {  this.paymentAmount = paymentAmount;  }  public Client getClient() {  return client;  }  public void setClient(Client client) {  this.client = client;  }  @Override  public String toString() {  return "Payment [paymentId=" + paymentId + ", paymentDate=" + paymentDate + ", paymentAmount=" + paymentAmount  + ", client=" + client + "]";  }  } |

1. **Defining an IPolicyService interface/abstract class with following methods to interact with database**

**Keeping the interfaces and implementation classes in package dao:**

1. createPolicy()

b. getPolicy()

c. getAllPolicies()

d. dupdatePolicy()

e. deletePolicy()

### ****DAO Interface****

**IPolicyService.java**

|  |
| --- |
| package hexa.org.dao;  import java.util.List;  import hexa.org.entity.Policy;  import hexa.org.exception.PolicyNotFoundException;  public interface IPolicyService {    boolean createPolicy(Policy policy);  Policy getPolicy(int policyId) throws PolicyNotFoundException;  List<Policy> getAllPolicies();  boolean updatePolicy(Policy policy);  boolean deletePolicy(int policyId) throws PolicyNotFoundException;  } |

1. **Define InsuranceServiceImpl class and implement all the methods InsuranceServiceImpl .**

**DAO Implementation**

InsuranceServiceImpl.java

|  |
| --- |
| package hexa.org.dao;  import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.util.ArrayList;  import java.util.List;  import hexa.org.entity.Policy;  import hexa.org.exception.PolicyNotFoundException;  import hexa.org.util.DBConnection;  public class InsuranceServiceImpl implements IPolicyService {    private Connection con;  public InsuranceServiceImpl() {  super();  con = DBConnection.getConnection();  }  List<Policy> policyList = new ArrayList<>();    @Override  public boolean createPolicy(Policy policy) {  boolean flag=false;  try {  PreparedStatement ps = con.prepareStatement("insert into policy(policyId,policyName,policyDetails) values(?,?,?)");  ps.setInt(1, policy.getPolicyId());  ps.setString(2, policy.getPolicyName());  ps.setString(3,policy.getPolicyDetails());  ps.executeUpdate();  flag=true;  }catch(SQLException se) {  System.out.println("Error while inserting policy");  se.printStackTrace();  }  return flag;  }  @Override  public Policy getPolicy(int policyId) throws PolicyNotFoundException {  Policy policy=null;  try {  PreparedStatement ps=con.prepareStatement("select \* from policy where policyId=?");  ps.setInt(1, policyId);  ResultSet rs = ps.executeQuery();    if(rs.next()) {  policy = new Policy(rs.getInt("PolicyId"),rs.getString("policyName"),rs.getString("policyDetails"));  }  else {  throw new PolicyNotFoundException("Policy not found");  }  }catch(SQLException se) {  System.out.println("Error while getting policy details");  se.printStackTrace();  }  return policy;  }  @Override  public List<Policy> getAllPolicies() {  Policy policy=null;  try {  PreparedStatement ps=con.prepareStatement("select \* from policy");  ResultSet rs=ps.executeQuery();    while(rs.next()) {  policy = new Policy(rs.getInt("policyId"),rs.getString("policyName"), rs.getString("policyDetails"));  policyList.add(policy);  }  }catch(SQLException se) {  System.out.println("Error while getting all policies");  se.printStackTrace();  }  return policyList;  }  @Override  public boolean updatePolicy(Policy policy) {  boolean flag=false;  try {  PreparedStatement ps=con.prepareStatement("update policy set policyName=?, policyDetails=? where policyId=?");  ps.setString(1, policy.getPolicyName());  ps.setString(2, policy.getPolicyDetails());  ps.setInt(3, policy.getPolicyId());  ps.executeUpdate();  flag=true;  }catch(SQLException se) {  System.out.println("Error while updating policy");  se.printStackTrace();  }  return flag;  }  @Override  public boolean deletePolicy(int policyId) throws PolicyNotFoundException {  boolean flag=false;  try {  PreparedStatement ps=con.prepareStatement("delete from policy where policyId=?");  ps.setInt(1, policyId);  int rows=ps.executeUpdate();  if(rows==0) {  throw new PolicyNotFoundException("Policy not found");  }  flag=true;  }catch(SQLException se) {  System.out.println("Error while deleting policy");  se.printStackTrace();    }  return flag;  }    } |

1. **Creating a utility class DBConnection in a package util with a static variable connection of Type Connection and a static method getConnection() which returns connection.**

**DBConnection.java**

|  |
| --- |
| package hexa.org.util;  import java.io.IOException;  import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.SQLException;  public class DBConnection {    private static final String fileName="db.properties";  public static Connection getConnection() {  Connection con=null;  String connString=null;  try {  connString=PropertyUtil.getPropertyString(fileName);  }catch(IOException e) {  System.out.println("Connection String Creation Failed...");  e.printStackTrace();  }  if(connString!=null) {  try {  con=DriverManager.getConnection(connString);  }catch(SQLException e) {  System.out.println("Error While Establishing DBConnection...");  e.printStackTrace();  }  }  return con;  }  **}** |

**Connection properties supplied in the connection string should be read from a property file which looks like this :**

**db.propeties**

|  |
| --- |
| protocol=jdbc:mysql:  system=localhost  port=3306  dbname=insurancedb  username=root  password=\*\*\*\*\*\*\*\*\*\* |

**Creating a utility class PropertyUtil which contains a static method named getPropertyString() which**

**reads a property fie containing connection details like hostname, dbname, username, password, port number and returns a connection string.**

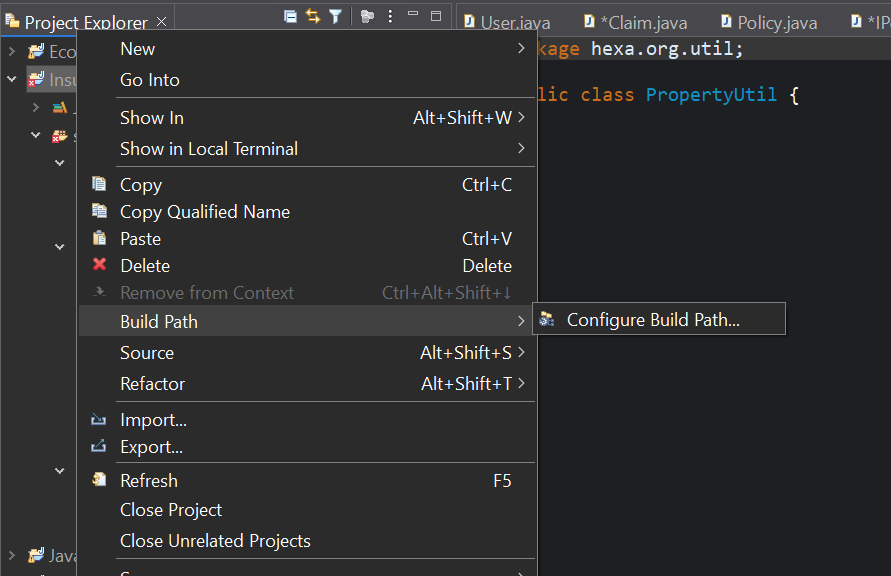
**PropertyUtil.java**

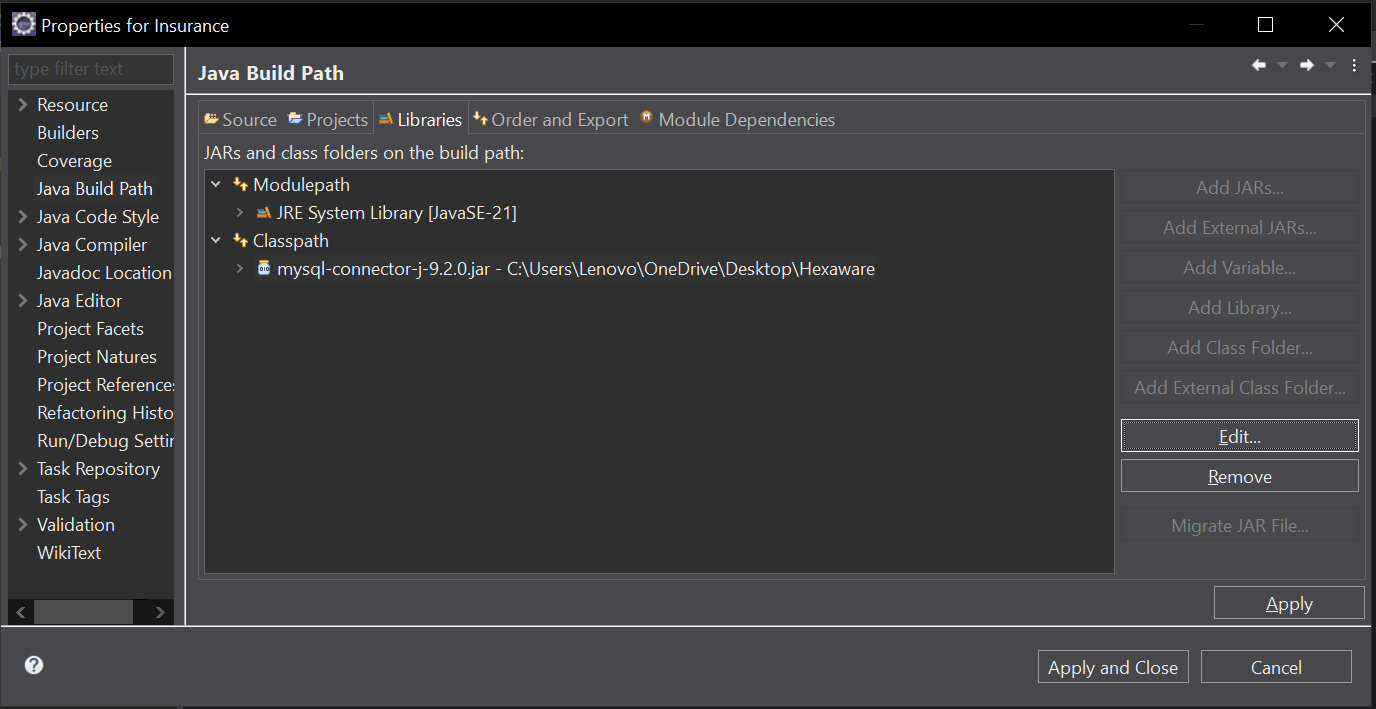
|  |
| --- |
| package hexa.org.util;  import java.io.FileInputStream;  import java.io.IOException;  import java.util.Properties;  public class PropertyUtil {  public static String getPropertyString(String fileName)throws IOException{  String connStr=null;  Properties props=new Properties();  FileInputStream fis=new FileInputStream(fileName);  props.load(fis);  String user=props.getProperty("username");  String password=props.getProperty("password");  String protocol=props.getProperty("protocol");  String system=props.getProperty("system");  String database=props.getProperty("dbname");  String port=props.getProperty("port");  connStr=protocol+"//"+system+":"+port+"/"+database+"?user="+user+"&password="+password;  return connStr;  }  } |

1. **Create the exceptions in package myexceptions Define the following custom exceptions and throw them in methods whenever needed. Handle all the exceptions in main method,**
2. **PolicyNotFoundException:**throw this exception when user enters an invalid patient number which doesn’t exist in db

|  |
| --- |
| package hexa.org.myexceptions;  public class PolicyNotFoundException extends Exception{  public PolicyNotFoundException(String message) {  super(message);  }  } |

1. **To enable Java to communicate with the MySQL database via JDBC, the MySQL JDBC driver (mysql-connector-j-9.2.0.jar) must be added to the classpath.**





1. **Create class named MainModule with main method in package main.**

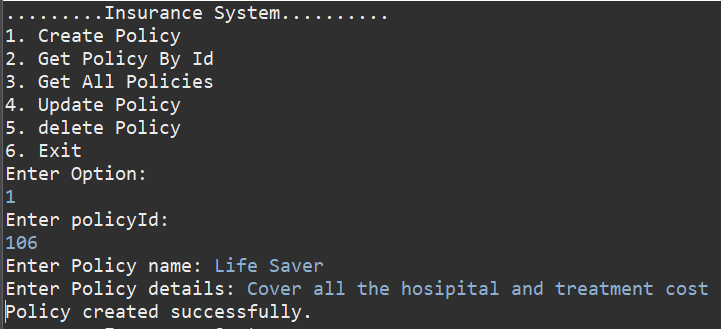
**Trigger all the methods in service implementation class.**

**MainModule.java**

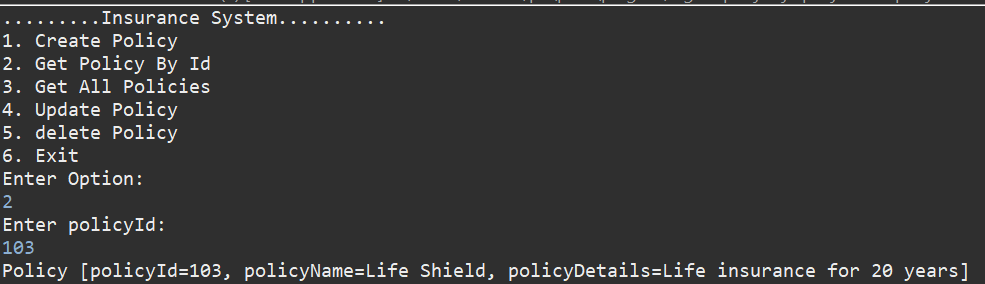
|  |
| --- |
| package hexa.org.main;  import java.util.List;  import java.util.Scanner;  import hexa.org.dao.InsuranceServiceImpl;  import hexa.org.entity.Policy;  import hexa.org.myexceptions.PolicyNotFoundException;  public class MainModule {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  InsuranceServiceImpl service = new InsuranceServiceImpl();    while(true) {  System.out.println(".........Insurance System..........");  System.out.println("1. Create Policy");  System.out.println("2. Get Policy By Id");  System.out.println("3. Get All Policies");  System.out.println("4. Update Policy");  System.out.println("5. delete Policy");  System.out.println("6. Exit");  System.out.println("Enter Option: ");  int option=sc.nextInt();    switch(option) {  case 1 -> {  System.out.println("Enter policyId: ");  int pid = sc.nextInt();  sc.nextLine();  System.out.print("Enter Policy name: ");  String name = sc.nextLine();  System.out.print("Enter Policy details: ");  String details = sc.nextLine();  Policy policy = new Policy(pid, name, details);  if(service.createPolicy(policy)) {  System.out.println("Policy created successfully.");  }else {  System.out.println("Policy creation failed");  }  }  case 2 -> {  System.out.println("Enter policyId:");  int id=sc.nextInt();  try {  Policy result=service.getPolicy(id);  System.out.println(result);  }catch(PolicyNotFoundException pe) {  System.out.println(pe.getMessage());  }  }  case 3 -> {  List<Policy> policies = service.getAllPolicies();  if(policies.isEmpty()) {  System.out.println("No policies found.");  }else {  for(Policy p: policies) {  System.out.println(p);  }  }  }  case 4 -> {  System.out.print("Enter policyId to update: ");  int upid = sc.nextInt();  sc.nextLine();  System.out.print("Enter New Policy name: ");  String newName = sc.nextLine();  System.out.print("Enter New Policy details: ");  String newDetails = sc.nextLine();  Policy updatedPolicy = new Policy(upid, newName, newDetails);  if (service.updatePolicy(updatedPolicy)) {  System.out.println("Policy updated successfully.");  } else {  System.out.println("Update failed.");  }      }  case 5 -> {  System.out.print("Enter PolicyId to delete: ");  int delid = sc.nextInt();  try {  if (service.deletePolicy(delid)) {  System.out.println("Policy deleted successfully.");  }  } catch (PolicyNotFoundException e) {  System.out.println(e.getMessage());  }  }  case 6 -> {  System.out.println("Exited...");  sc.close();  System.exit(0);  }  default -> {  System.out.println("Invalid option");  }  }  }  }  } |

1. **OUTPUT:**

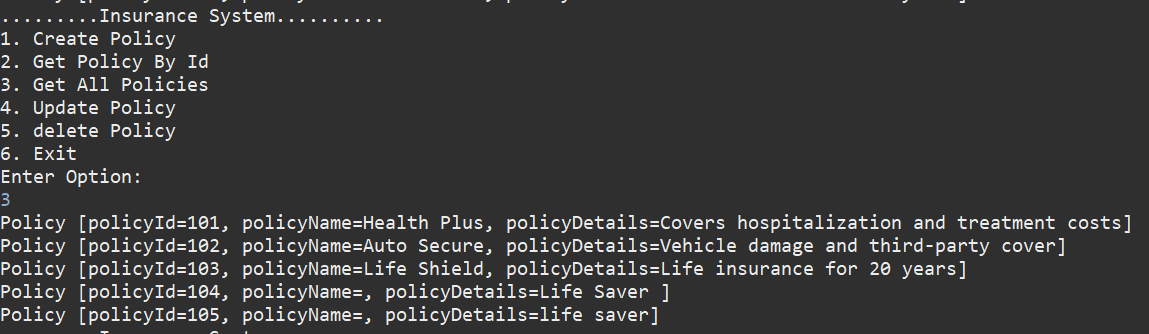
**When option:1**



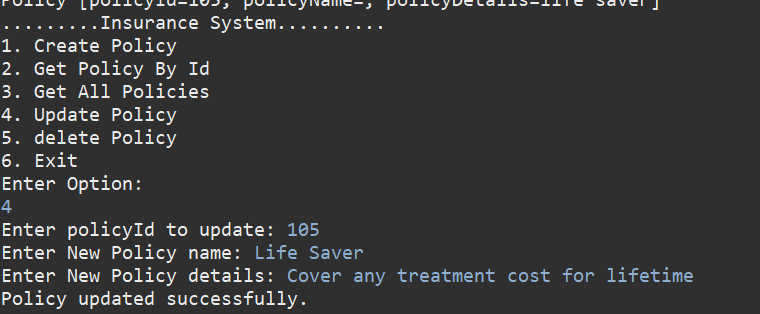
**When option:2**



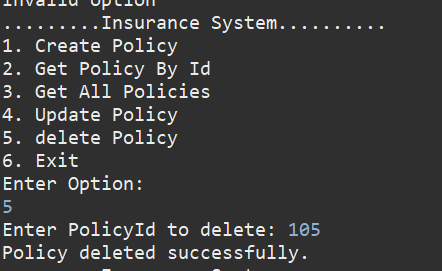
**When option:3**



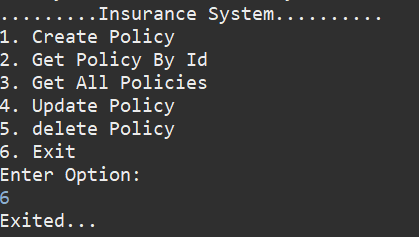
**When option:4**



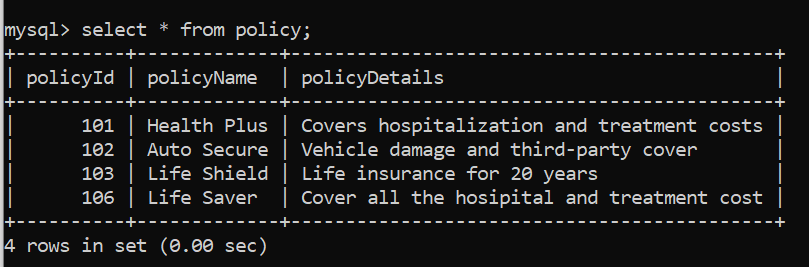
**When option:5**



**When option:6**



**Verification of Database Update from Java Application**



GitHub link: <https://github.com/sherinsandra03/Insurance/tree/main/Insurance1>