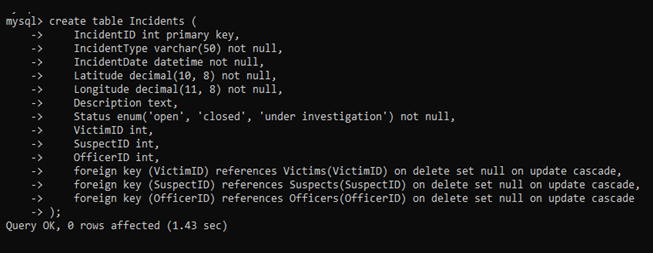
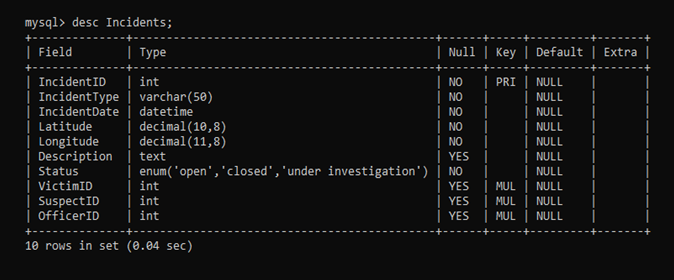
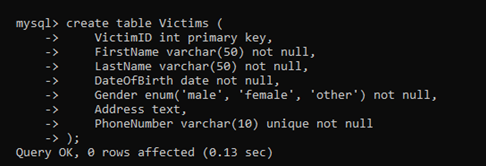
Crime Reporting System – Schema Design

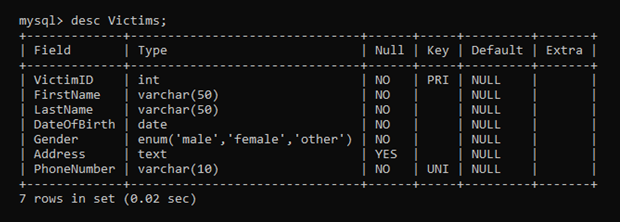
1. **Incidents Table:**



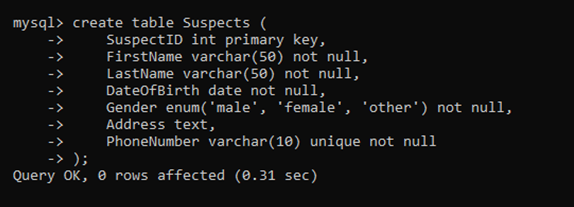


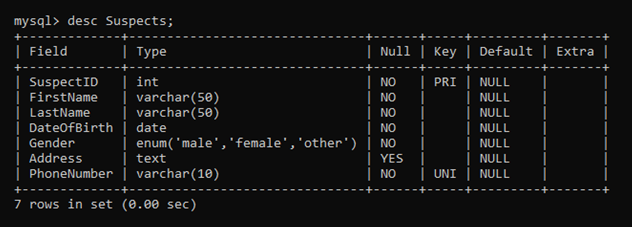
1. **Victims Table:**



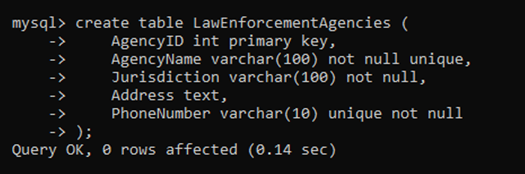


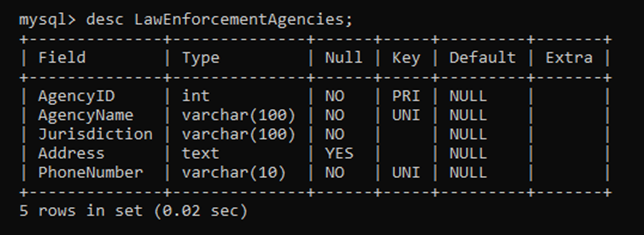
1. **Suspects Table:**



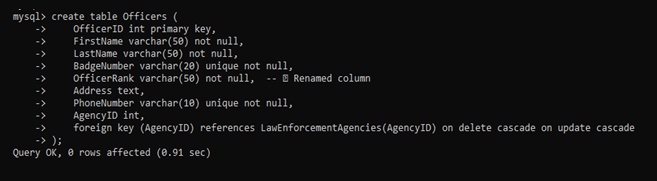


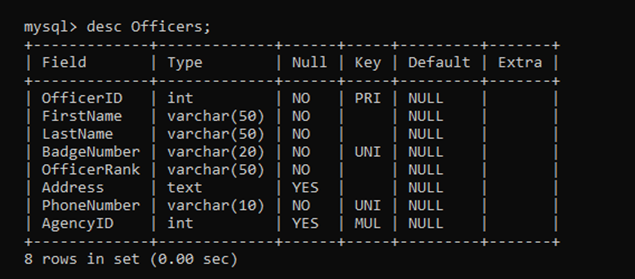
1. **Law Enforcement Agencies Table:**



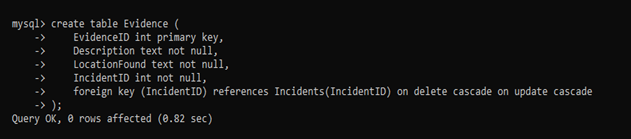


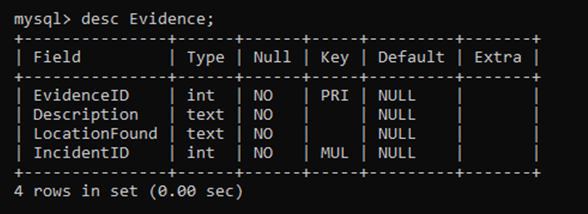
1. **Officers Table:**



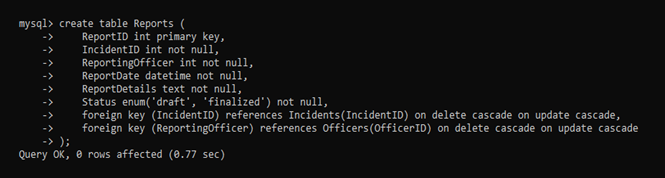


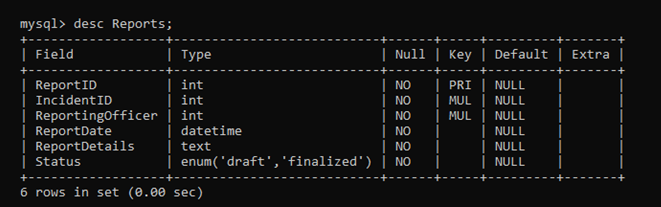
1. **Evidence Table:**



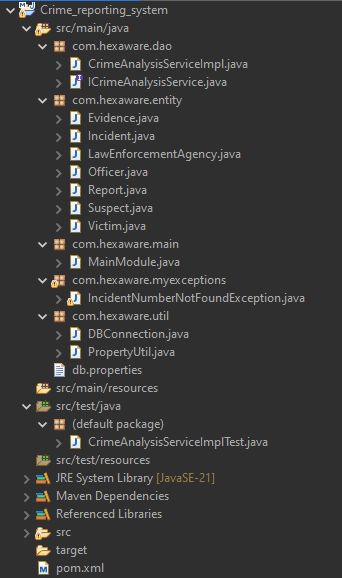


1. **Reports Table:**





**Directory structure in Eclipse:**



**Create the model/entity classes corresponding to the schema within package entity with variables declared private, constructors(default and parametrized) and getters,setters )**

**Evidence.java**

package com.hexaware.entity;

public class Evidence {

private int evidenceID;

private String description;

private String locationFound;

private int incidentID;

// Default constructor

public Evidence() {

super();

}

// Parameterized constructor

public Evidence(int evidenceID, String description, String locationFound, int incidentID) {

super();

this.evidenceID = evidenceID;

this.description = description;

this.locationFound = locationFound;

this.incidentID = incidentID;

}

// Getters and setters

public int getEvidenceID() {

return evidenceID;

}

public void setEvidenceID(int evidenceID) {

this.evidenceID = evidenceID;

}

public String getDescription() {

return description;

}

public void setDescription(String description) {

this.description = description;

}

public String getLocationFound() {

return locationFound;

}

public void setLocationFound(String locationFound) {

this.locationFound = locationFound;

}

public int getIncidentID() {

return incidentID;

}

public void setIncidentID(int incidentID) {

this.incidentID = incidentID;

}

}

**Incident.java**

package com.hexaware.entity;

public class Incident {

private int incidentID;

private String incidentType;

private String incidentDate;

private double latitude;

private double longitude;

private String description;

private String status;

private int victimID;

private int suspectID;

private int officerID;

// Default constructor

public Incident() {

super();

}

// Parameterized constructor

public Incident(int incidentID, String incidentType, String incidentDate, double latitude, double longitude,

String description, String status, int victimID, int suspectID, int officerID) {

super();

this.incidentID = incidentID;

this.incidentType = incidentType;

this.incidentDate = incidentDate;

this.latitude = latitude;

this.longitude = longitude;

this.description = description;

this.status = status;

this.victimID = victimID;

this.suspectID = suspectID;

this.officerID = officerID;

}

*@Override*

public String toString() {

return "incidentID=" + incidentID +

", type='" + incidentType + '\'' +

", date='" + incidentDate + '\'' +

", lat=" + latitude +

", long=" + longitude +

", description='" + description + '\'' +

", status='" + status + '\'' +

", victimID=" + victimID +

", suspectID=" + suspectID +

", officerID=" + officerID +

'}';

}

// Getters and setters

public int getIncidentID() {

return incidentID;

}

public void setIncidentID(int incidentID) {

this.incidentID = incidentID;

}

public String getIncidentType() {

return incidentType;

}

public void setIncidentType(String incidentType) {

this.incidentType = incidentType;

}

public String getIncidentDate() {

return incidentDate;

}

public void setIncidentDate(String incidentDate) {

this.incidentDate = incidentDate;

}

public double getLatitude() {

return latitude;

}

public void setLatitude(double latitude) {

this.latitude = latitude;

}

public double getLongitude() {

return longitude;

}

public void setLongitude(double longitude) {

this.longitude = longitude;

}

public String getDescription() {

return description;

}

public void setDescription(String description) {

this.description = description;

}

public String getStatus() {

return status;

}

public void setStatus(String status) {

this.status = status;

}

public int getVictimID() {

return victimID;

}

public void setVictimID(int victimID) {

this.victimID = victimID;

}

public int getSuspectID() {

return suspectID;

}

public void setSuspectID(int suspectID) {

this.suspectID = suspectID;

}

public int getOfficerID() {

return officerID;

}

public void setOfficerID(int officerID) {

this.officerID = officerID;

}

private String tempReportDate;

private String tempReportDetails;

private String tempReportStatus;

public String getTempReportDate() {

return tempReportDate;

}

public void setTempReportDate(String tempReportDate) {

this.tempReportDate = tempReportDate;

}

public String getTempReportDetails() {

return tempReportDetails;

}

public void setTempReportDetails(String tempReportDetails) {

this.tempReportDetails = tempReportDetails;

}

public String getTempReportStatus() {

return tempReportStatus;

}

public void setTempReportStatus(String tempReportStatus) {

this.tempReportStatus = tempReportStatus;

}

}

**LawEnforcementAgency.java**

package com.hexaware.entity;

public class LawEnforcementAgency {

private int agencyID;

private String agencyName;

private String jurisdiction;

private String contactInfo;

// Default constructor

public LawEnforcementAgency() {

super();

}

// Parameterized constructor

public LawEnforcementAgency(int agencyID, String agencyName, String jurisdiction, String contactInfo) {

super();

this.agencyID = agencyID;

this.agencyName = agencyName;

this.jurisdiction = jurisdiction;

this.contactInfo = contactInfo;

}

// Getters and setters

public int getAgencyID() {

return agencyID;

}

public void setAgencyID(int agencyID) {

this.agencyID = agencyID;

}

public String getAgencyName() {

return agencyName;

}

public void setAgencyName(String agencyName) {

this.agencyName = agencyName;

}

public String getJurisdiction() {

return jurisdiction;

}

public void setJurisdiction(String jurisdiction) {

this.jurisdiction = jurisdiction;

}

public String getContactInfo() {

return contactInfo;

}

public void setContactInfo(String contactInfo) {

this.contactInfo = contactInfo;

}

}

**Officer.java**

package com.hexaware.entity;

public class Officer {

private int officerID;

private String firstName;

private String lastName;

private String badgeNumber;

private String rank;

private String contactInfo;

private int agencyID;

// Default constructor

public Officer() {

super();

}

// Parameterized constructor

public Officer(int officerID, String firstName, String lastName, String badgeNumber, String rank,

String contactInfo, int agencyID) {

super();

this.officerID = officerID;

this.firstName = firstName;

this.lastName = lastName;

this.badgeNumber = badgeNumber;

this.rank = rank;

this.contactInfo = contactInfo;

this.agencyID = agencyID;

}

// Getters and setters

public int getOfficerID() {

return officerID;

}

public void setOfficerID(int officerID) {

this.officerID = officerID;

}

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public String getBadgeNumber() {

return badgeNumber;

}

public void setBadgeNumber(String badgeNumber) {

this.badgeNumber = badgeNumber;

}

public String getRank() {

return rank;

}

public void setRank(String rank) {

this.rank = rank;

}

public String getContactInfo() {

return contactInfo;

}

public void setContactInfo(String contactInfo) {

this.contactInfo = contactInfo;

}

public int getAgencyID() {

return agencyID;

}

public void setAgencyID(int agencyID) {

this.agencyID = agencyID;

}

**Report.java**

package com.hexaware.entity;

public class Report {

private int reportID;

private int incidentID;

private int reportingOfficer;

private String reportDate;

private String reportDetails;

private String status;

// Default constructor

public Report() {

super();

}

// Parameterized constructor

public Report(int reportID, int incidentID, int reportingOfficer, String reportDate, String reportDetails,

String status) {

super();

this.reportID = reportID;

this.incidentID = incidentID;

this.reportingOfficer = reportingOfficer;

this.reportDate = reportDate;

this.reportDetails = reportDetails;

this.status = status;

}

*@Override*

public String toString() {

return "incidentID=" + incidentID +

", reportingOfficer=" + reportingOfficer +

", reportDate='" + reportDate + '\'' +

", details='" + reportDetails + '\'' +

", status='" + status + '\'' +

'}';

}

// Getters and setters

public int getReportID() {

return reportID;

}

public void setReportID(int reportID) {

this.reportID = reportID;

}

public int getIncidentID() {

return incidentID;

}

public void setIncidentID(int incidentID) {

this.incidentID = incidentID;

}

public int getReportingOfficer() {

return reportingOfficer;

}

public void setReportingOfficer(int reportingOfficer) {

this.reportingOfficer = reportingOfficer;

}

public String getReportDate() {

return reportDate;

}

public void setReportDate(String reportDate) {

this.reportDate = reportDate;

}

public String getReportDetails() {

return reportDetails;

}

public void setReportDetails(String reportDetails) {

this.reportDetails = reportDetails;

}

public String getStatus() {

return status;

}

public void setStatus(String status) {

this.status = status;

}

}

**Suspect.java**package com.hexaware.entity;

public class Suspect {

private int suspectID;

private String firstName;

private String lastName;

private String dateOfBirth;

private String gender;

private String contactInfo;

// Default constructor

public Suspect() {

super();

}

// Parameterized constructor

public Suspect(int suspectID, String firstName, String lastName, String dateOfBirth, String gender,

String contactInfo) {

super();

this.suspectID = suspectID;

this.firstName = firstName;

this.lastName = lastName;

this.dateOfBirth = dateOfBirth;

this.gender = gender;

this.contactInfo = contactInfo;

}

// Getters and setters

public int getSuspectID() {

return suspectID;

}

public void setSuspectID(int suspectID) {

this.suspectID = suspectID;

}

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public String getDateOfBirth() {

return dateOfBirth;

}

public void setDateOfBirth(String dateOfBirth) {

this.dateOfBirth = dateOfBirth;

}

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

public String getContactInfo() {

return contactInfo;

}

public void setContactInfo(String contactInfo) {

this.contactInfo = contactInfo;

}

}

**Victim.java**

package com.hexaware.entity;

public class Victim {

private int victimID;

private String firstName;

private String lastName;

private String dateOfBirth;

private String gender;

private String contactInfo;

// Default constructor

public Victim() {

super();

}

// Parameterized constructor

public Victim(int victimID, String firstName, String lastName, String dateOfBirth, String gender, String contactInfo) {

super();

this.victimID = victimID;

this.firstName = firstName;

this.lastName = lastName;

this.dateOfBirth = dateOfBirth;

this.gender = gender;

this.contactInfo = contactInfo;

}

// Getters and setters

public int getVictimID() {

return victimID;

}

public void setVictimID(int victimID) {

this.victimID = victimID;

}

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public String getDateOfBirth() {

return dateOfBirth;

}

public void setDateOfBirth(String dateOfBirth) {

this.dateOfBirth = dateOfBirth;

}

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

public String getContactInfo() {

return contactInfo;

}

public void setContactInfo(String contactInfo) {

this.contactInfo = contactInfo;

}

}

**Service Provider Interface/Abstract class**

• Keep the interfaces and implementation classes in package dao

Create ICrimeAnalysisService Interface/abstract classs with the following methods

// Create a new incident

createIncident();

parameters- Incident object

return type Boolean

// Update the status of an incident

updateIncidentStatus();

parameters- Status object,incidentid

return type Boolean

// Get a list of incidents within a date range

getIncidentsInDateRange();

parameters- startDate, endDate

return type Collection of Incident objects

// Search for incidents based on various criteria

searchIncidents(IncidentType criteria);

parameters- IncidentType object

return type Collection of Incident objects

// Generate incident reports

generateIncidentReport();

parameters- Incident object

return type Report object

**ICrimeAnalysisService.java (Interface)**

package com.hexaware.dao;

import java.util.Collection;

import com.hexaware.entity.Incident;

import com.hexaware.entity.Report;

import java.time.LocalDate;

import com.hexaware.myexceptions.IncidentNumberNotFoundException;

public interface ICrimeAnalysisService {

boolean createIncident(Incident incident);

boolean updateIncidentStatus(String status, int incidentID) throws IncidentNumberNotFoundException;

Collection<Incident> getIncidentsInDateRange(LocalDate startDate, LocalDate endDate);

Collection<Incident> searchIncidents(String incidentType);

Report generateIncidentReport(Incident incident);

}

**Connect your application to the SQL database:**

1. Write code to establish a connection to your SQL database.

Create a utility class DBConnection in a package util with a static variable connection of Type

Connection and a static method getConnection() which returns connection.

Connection properties supplied in the connection string should be read from a property file.

Create 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

**db.properties**

host=localhost

port=3306

dbname=crime\_reporting\_system

username=root

password=@MrBTS07691234

**PropertyUtil.java**  
package com.hexaware.util;

import java.io.InputStream;

import java.util.Properties;

public class PropertyUtil {

public static String getPropertyString() {

Properties props = new Properties();

try {

InputStream input = PropertyUtil.class.getClassLoader().getResourceAsStream("db.properties");

if (input == null) {

System.out.println("db.properties file not found in classpath!");

return null;

}

props.load(input);

} catch (Exception e) {

e.printStackTrace();

return null;

}

String host = props.getProperty("host");

String port = props.getProperty("port");

String dbname = props.getProperty("dbname");

String username = props.getProperty("username");

String password = props.getProperty("password");

return "jdbc:mysql://" + host + ":" + port + "/" + dbname + "?user=" + username + "&password=" + password;

}

}

**DBConnection.java**package com.hexaware.util;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class DBConnection {

private static Connection connection;

public static Connection getConnection() {

if(connection == null) {

try {

String connectionUrl = PropertyUtil.getPropertyString();

if(connectionUrl != null) {

connection = DriverManager.getConnection(connectionUrl);

System.out.println("Database connected successfully!");

}

else {

System.out.println("Connection string is null.");

}

}

catch(SQLException e) {

System.out.println("Failed to connect to database");

e.printStackTrace();

}

}

return connection;

}

}

**7: Service implementation**

1. Create a Service class CrimeAnalysisServiceImpl in package dao with a static variable named

connection of type Connection which can be assigned in the constructor by invoking the

getConnection() method in DBConnection class

2. Provide implementation for all the methods in the interface/abstract class

**CrimeAnalysisServiceImpl.java (Implementation class):**

package com.hexaware.dao;

import java.time.format.DateTimeFormatter;

import java.time.format.DateTimeParseException;

import java.sql.\*;

import java.time.LocalDate;

import java.util.ArrayList;

import java.util.Collection;

import com.hexaware.entity.Incident;

import com.hexaware.entity.Report;

import com.hexaware.util.DBConnection;

import com.hexaware.myexceptions.IncidentNumberNotFoundException;

public class CrimeAnalysisServiceImpl implements ICrimeAnalysisService {

private static Connection connection;

public CrimeAnalysisServiceImpl() {

connection = DBConnection.getConnection();

}

public CrimeAnalysisServiceImpl(Connection conn) {

connection = conn;

}

@Override

public boolean createIncident(Incident incident) {

String query = "INSERT INTO Incidents (incidentType, incidentDate, latitude, longitude, description, status, victimID, suspectID, officerID) " +

"VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)";

try (PreparedStatement ps = connection.prepareStatement(query)) {

// Check date format strictly

DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");

LocalDate date = LocalDate.parse(incident.getIncidentDate(), formatter);

ps.setString(1, incident.getIncidentType());

ps.setDate(2, Date.valueOf(date));

ps.setDouble(3, incident.getLatitude());

ps.setDouble(4, incident.getLongitude());

ps.setString(5, incident.getDescription());

ps.setString(6, incident.getStatus());

ps.setInt(7, incident.getVictimID());

ps.setInt(8, incident.getSuspectID());

ps.setInt(9, incident.getOfficerID());

return ps.executeUpdate() > 0;

} catch (DateTimeParseException e) {

System.err.println("Invalid date format. Expected yyyy-MM-dd. Got: " + incident.getIncidentDate());

} catch (SQLException e) {

e.printStackTrace();

}

return false;

}

@Override

public boolean updateIncidentStatus(String status, int incidentId) throws IncidentNumberNotFoundException {

String query = "UPDATE Incidents SET status = ? WHERE incidentID = ?";

try (PreparedStatement ps = connection.prepareStatement(query)) {

ps.setString(1, status);

ps.setInt(2, incidentId);

int rowsUpdated = ps.executeUpdate();

if (rowsUpdated == 0) {

throw new IncidentNumberNotFoundException("Incident ID " + incidentId + " not found in database.");

}

return true;

} catch (SQLException e) {

e.printStackTrace();

}

return false;

}

@Override

public Collection<Incident> getIncidentsInDateRange(LocalDate startDate, LocalDate endDate) {

Collection<Incident> incidents = new ArrayList<>();

String query = "SELECT \* FROM Incidents WHERE incidentDate BETWEEN ? AND ?";

try (PreparedStatement ps = connection.prepareStatement(query)) {

ps.setDate(1, Date.valueOf(startDate));

ps.setDate(2, Date.valueOf(endDate));

ResultSet rs = ps.executeQuery();

while (rs.next()) {

incidents.add(mapResultSetToIncident(rs));

}

} catch (SQLException e) {

e.printStackTrace();

}

return incidents;

}

@Override

public Collection<Incident> searchIncidents(String incidentType) {

Collection<Incident> incidents = new ArrayList<>();

String query = "SELECT \* FROM Incidents WHERE incidentType = ?";

try (PreparedStatement ps = connection.prepareStatement(query)) {

ps.setString(1, incidentType);

ResultSet rs = ps.executeQuery();

while (rs.next()) {

incidents.add(mapResultSetToIncident(rs));

}

} catch (SQLException e) {

e.printStackTrace();

}

return incidents;

}

@Override

public Report generateIncidentReport(Incident incident) {

String insertQuery = "INSERT INTO Reports (IncidentID, ReportingOfficer, ReportDate, ReportDetails, Status) VALUES (?, ?, ?, ?, ?)";

Report report = new Report();

try (PreparedStatement ps = connection.prepareStatement(insertQuery, Statement.RETURN\_GENERATED\_KEYS)) {

ps.setInt(1, incident.getIncidentID());

ps.setInt(2, incident.getOfficerID());

ps.setString(3, incident.getTempReportDate());

ps.setString(4, incident.getTempReportDetails());

ps.setString(5, incident.getTempReportStatus());

int rowsInserted = ps.executeUpdate();

if (rowsInserted > 0) {

ResultSet generatedKeys = ps.getGeneratedKeys();

if (generatedKeys.next()) {

int generatedReportID = generatedKeys.getInt(1);

report.setReportID(generatedReportID);

report.setIncidentID(incident.getIncidentID());

report.setReportingOfficer(incident.getOfficerID());

report.setReportDate(incident.getTempReportDate());

report.setReportDetails(incident.getTempReportDetails());

report.setStatus(incident.getTempReportStatus());

}

}

} catch (SQLException e) {

e.printStackTrace();

return null;

}

return report;

}

public Incident getIncidentById(int id) {

String query = "SELECT \* FROM Incidents WHERE incidentID = ?";

try (PreparedStatement ps = connection.prepareStatement(query)) {

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if (rs.next()) {

return mapResultSetToIncident(rs);

}

} catch (SQLException e) {

e.printStackTrace();

}

return null;

}

private Incident mapResultSetToIncident(ResultSet rs) throws SQLException {

return new Incident(

rs.getInt("incidentID"),

rs.getString("incidentType"),

rs.getDate("incidentDate").toLocalDate().toString(), // convert to LocalDate then to String

rs.getDouble("latitude"),

rs.getDouble("longitude"),

rs.getString("description"),

rs.getString("status"),

rs.getInt("victimID"),

rs.getInt("suspectID"),

rs.getInt("officerID")

);

}

}

**8: Exception Handling:**

Create the exceptions in package com.hexaware.myexceptions

Define the following custom exceptions and throw them in methods whenever needed. Handle all the

exceptions in main method,

1. IncidentNumberNotFoundException :throw this exception when user enters an invalid patient

number which doesn’t exist in db

**IncidentNumberNotFoundException.java**

package com.hexaware.myexceptions;

public class IncidentNumberNotFoundException extends Exception {

public IncidentNumberNotFoundException(String message) {

super(message);

}

}

**9. Main Method**

Create class named MainModule with main method in main package.

Trigger all the methods in service implementation class.

**MainModule.java**

package com.hexaware.main;

import java.util.Scanner;

import java.time.LocalDate;

import java.util.Collection;

import com.hexaware.dao.CrimeAnalysisServiceImpl;

import com.hexaware.entity.Incident;

import com.hexaware.entity.Report;

import com.hexaware.myexceptions.IncidentNumberNotFoundException;

public class MainModule {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

CrimeAnalysisServiceImpl service = new CrimeAnalysisServiceImpl();

int choice;

do {

System.out.println("\n====== Crime Analysis System ======");

System.out.println("1. Create Incident");

System.out.println("2. Update Incident Status");

System.out.println("3. View Incidents in Date Range");

System.out.println("4. Search Incidents by Type");

System.out.println("5. Generate Report for Incident");

System.out.println("6. Exit");

System.out.print("Enter your choice: ");

choice = sc.nextInt();

sc.nextLine(); // consume newline

try {

switch (choice) {

case 1:

Incident incident = new Incident();

System.out.print("Enter Incident Type: ");

incident.setIncidentType(sc.nextLine());

System.out.print("Enter Incident Date (yyyy-mm-dd): ");

incident.setIncidentDate(sc.nextLine());

System.out.print("Enter Latitude: ");

incident.setLatitude(sc.nextDouble());

System.out.print("Enter Longitude: ");

incident.setLongitude(sc.nextDouble());

sc.nextLine(); // consume newline

System.out.print("Enter Description: ");

incident.setDescription(sc.nextLine());

System.out.print("Enter Status: ");

incident.setStatus(sc.nextLine());

System.out.print("Enter Victim ID: ");

incident.setVictimID(sc.nextInt());

System.out.print("Enter Suspect ID: ");

incident.setSuspectID(sc.nextInt());

System.out.print("Enter Officer ID: ");

incident.setOfficerID(sc.nextInt());

boolean created = service.createIncident(incident);

System.out.println(created ? "Incident created successfully." : "Failed to create incident.");

break;

case 2:

System.out.print("Enter Incident ID to update: ");

int incidentId = sc.nextInt();

sc.nextLine(); // consume newline

System.out.print("Enter new status: ");

String newStatus = sc.nextLine();

boolean updated = service.updateIncidentStatus(newStatus, incidentId);

if (updated) {

System.out.println("Incident updated successfully.");

} else {

System.out.println("Incident not found or update failed.");

}

break;

case 3:

System.out.print("Enter Start Date (yyyy-mm-dd): ");

LocalDate start = LocalDate.parse(sc.nextLine());

System.out.print("Enter End Date (yyyy-mm-dd): ");

LocalDate end = LocalDate.parse(sc.nextLine());

Collection<Incident> incidentsInRange = service.getIncidentsInDateRange(start, end);

if (incidentsInRange.isEmpty()) {

System.out.println("No incidents found in the given date range.");

} else {

incidentsInRange.forEach(System.out::println);

}

break;

case 4:

System.out.print("Enter Incident Type to search: ");

String type = sc.nextLine();

Collection<Incident> searchResults = service.searchIncidents(type);

if (searchResults.isEmpty()) {

System.out.println("No incidents found of type: " + type);

} else {

searchResults.forEach(System.out::println);

}

break;

case 5:

System.out.print("Enter Incident ID to generate report: ");

int reportIncidentId = sc.nextInt();

sc.nextLine(); // consume newline

Incident selected = service.getIncidentById(reportIncidentId);

if (selected != null) {

// Take report input here and set values in the incident

System.out.print("Enter Report Date (YYYY-MM-DD): ");

String reportDate = sc.nextLine();

System.out.print("Enter Report Details: ");

String reportDetails = sc.nextLine();

System.out.print("Enter Report Status (e.g., draft/finalized): ");

String reportStatus = sc.nextLine();

// Set temporarily into Incident for transfer

selected.setTempReportDate(reportDate);

selected.setTempReportDetails(reportDetails);

selected.setTempReportStatus(reportStatus);

Report report = service.generateIncidentReport(selected);

if (report != null) {

System.out.println("Report successfully generated and added to database.");

System.out.println(report);

} else {

System.out.println("Failed to generate and store report.");

}

} else {

throw new IncidentNumberNotFoundException("Incident ID " + reportIncidentId + " not found.");

}

break;

case 6:

System.out.println("Exiting application.");

break;

default:

System.out.println("Invalid choice!");

}

} catch (IncidentNumberNotFoundException e) {

System.err.println("Error: " + e.getMessage());

} catch (Exception e) {

System.err.println("Unexpected Error: " + e.getMessage());

}

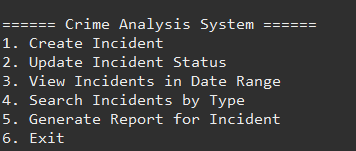
} while (choice != 6);

sc.close();

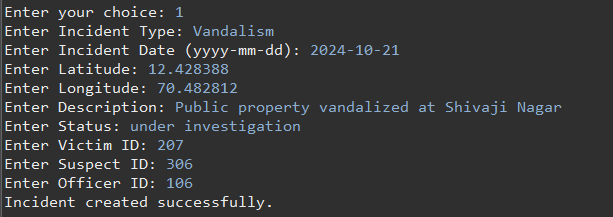
}

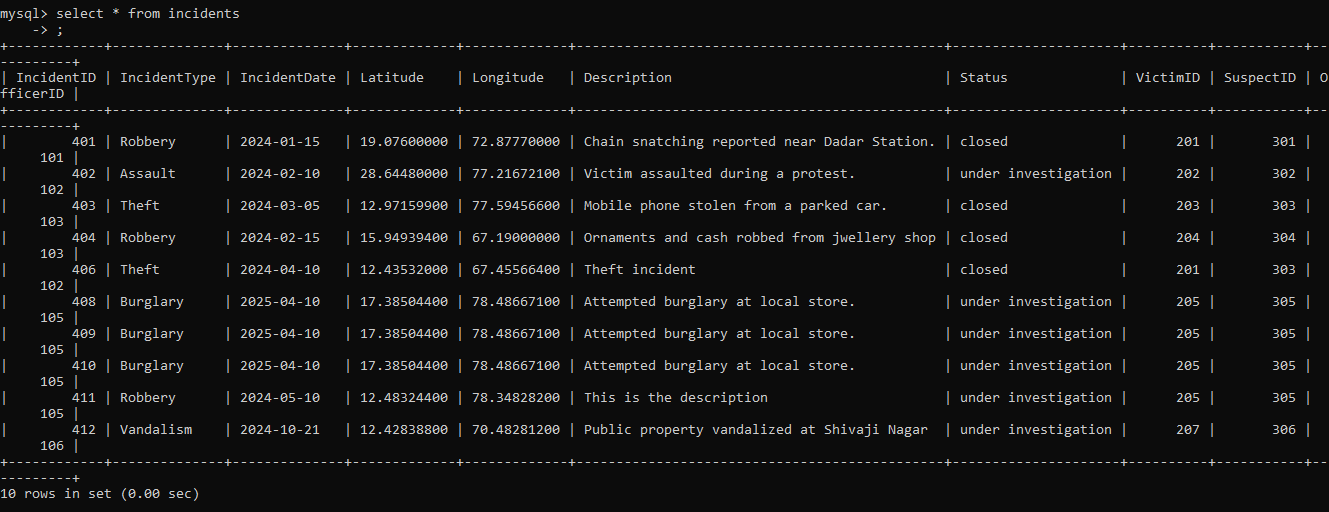
}

**Output:**

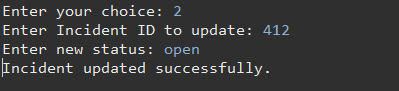
****

**Option 1: Create incident:**

****

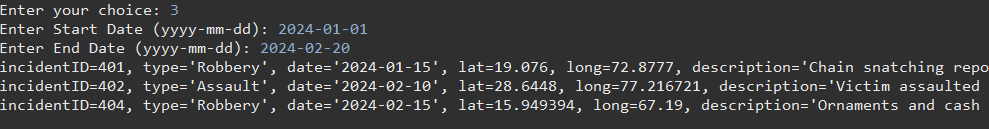
****

**Option 2: Update incident status**

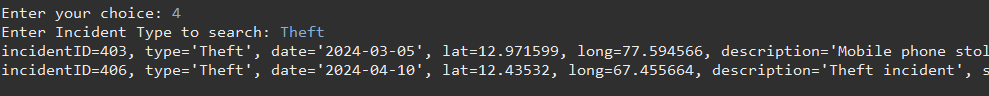
****

****

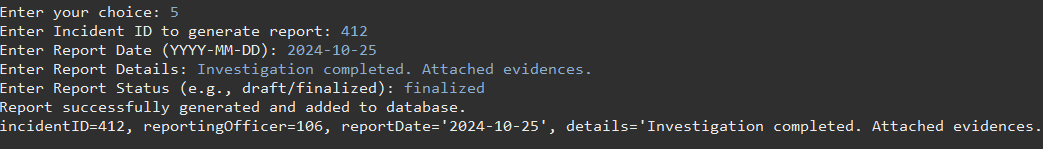
**Option 3: View Incidents in date range**

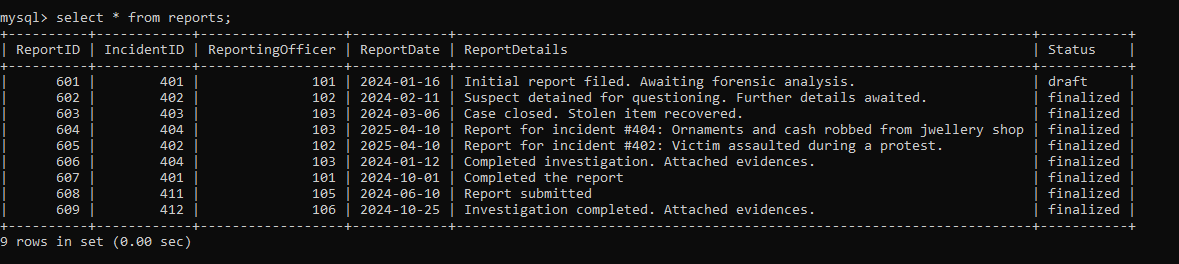
****

**Option 4: Search incidents by type**

****

**Option 5: Generate report by incident**

****

****

**10. Unit Testing**

Creating JUnit test cases for a Crime Analysis and Reporting System is essential to ensure the

correctness and reliability of your system. Below are some example questions to guide the creation

of JUnit test cases for various components of the system:

1. Incident Creation:

• Does the createIncident method correctly create an incident with the provided

attributes?

• Are the attributes of the created incident accurate?

2. Incident Status Update:

• Does the updateIncidentStatus method effectively update the status of an incident?

• Does it handle invalid status updates appropriately?

**CrimeAnalysisServiceImplTest.java:**

import static org.junit.jupiter.api.Assertions.\*;

import com.hexaware.entity.Incident;

import com.hexaware.myexceptions.IncidentNumberNotFoundException;

import com.hexaware.dao.CrimeAnalysisServiceImpl;

import com.hexaware.util.DBConnection;

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.Test;

import java.sql.Connection;

public class CrimeAnalysisServiceImplTest {

private CrimeAnalysisServiceImpl service;

*@BeforeEach*

public void setUp() {

Connection conn = DBConnection.*getConnection*();

service = new CrimeAnalysisServiceImpl(conn);

}

// Test 1: Create a valid incident

*@Test*

public void testCreateIncident\_Success() {

Incident incident = new Incident();

incident.setIncidentType("Burglary");

incident.setIncidentDate("2025-04-10");

incident.setLatitude(17.385044);

incident.setLongitude(78.486671);

incident.setDescription("Attempted burglary at local store.");

incident.setStatus("under investigation");

incident.setVictimID(205);

incident.setSuspectID(305);

incident.setOfficerID(105);

boolean result = service.createIncident(incident);

if (result) {

System.***out***.println("Test Passed: Incident created successfully.");

} else {

System.***out***.println("Test Failed: Incident creation failed.");

}

*assertTrue*(result, "Incident should be successfully created");

}

// Test 2: Create an incident with invalid date format

*@Test*

public void testCreateIncident\_InvalidDate() {

Incident incident = new Incident();

incident.setIncidentType("Vandalism");

incident.setIncidentDate("10-04-2025"); // Invalid format

incident.setLatitude(19.2183);

incident.setLongitude(72.9781);

incident.setDescription("Graffiti on school wall.");

incident.setStatus("under investigation");

incident.setVictimID(210);

incident.setSuspectID(310);

incident.setOfficerID(110);

boolean result = service.createIncident(incident);

if (!result) {

System.***out***.println("Test Passed: Incident creation failed due to invalid date format.");

} else {

System.***out***.println("Test Failed: Incident created with invalid date format.");

}

*assertFalse*(result, "Incident creation should fail due to invalid date");

}

// Test 3: Update status of a valid incident

*@Test*

public void testUpdateIncidentStatus\_Success() throws IncidentNumberNotFoundException {

boolean result = service.updateIncidentStatus("closed", 401);

if (result) {

System.***out***.println("Test Passed: Incident status updated successfully.");

} else {

System.***out***.println("Test Failed: Failed to update incident status.");

}

*assertTrue*(result, "Status should be updated for valid incident ID");

}

// Test 4: Update status of an invalid incident ID

*@Test*

public void testUpdateIncidentStatus\_InvalidID() {

try {

service.updateIncidentStatus("closed", 9999);

System.***out***.println("Test Failed: Expected exception for invalid incident ID was not thrown.");

*fail*("Expected IncidentNumberNotFoundException");

} catch (IncidentNumberNotFoundException e) {

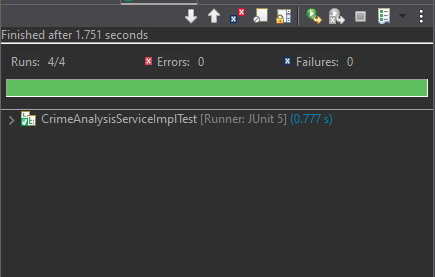
System.***out***.println("Test Passed: Correctly threw IncidentNumberNotFoundException for invalid ID 9999.");

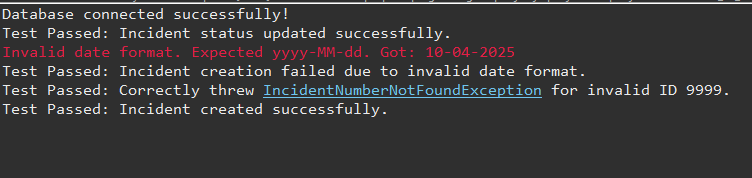
}

}

}

**Output:**

****

****