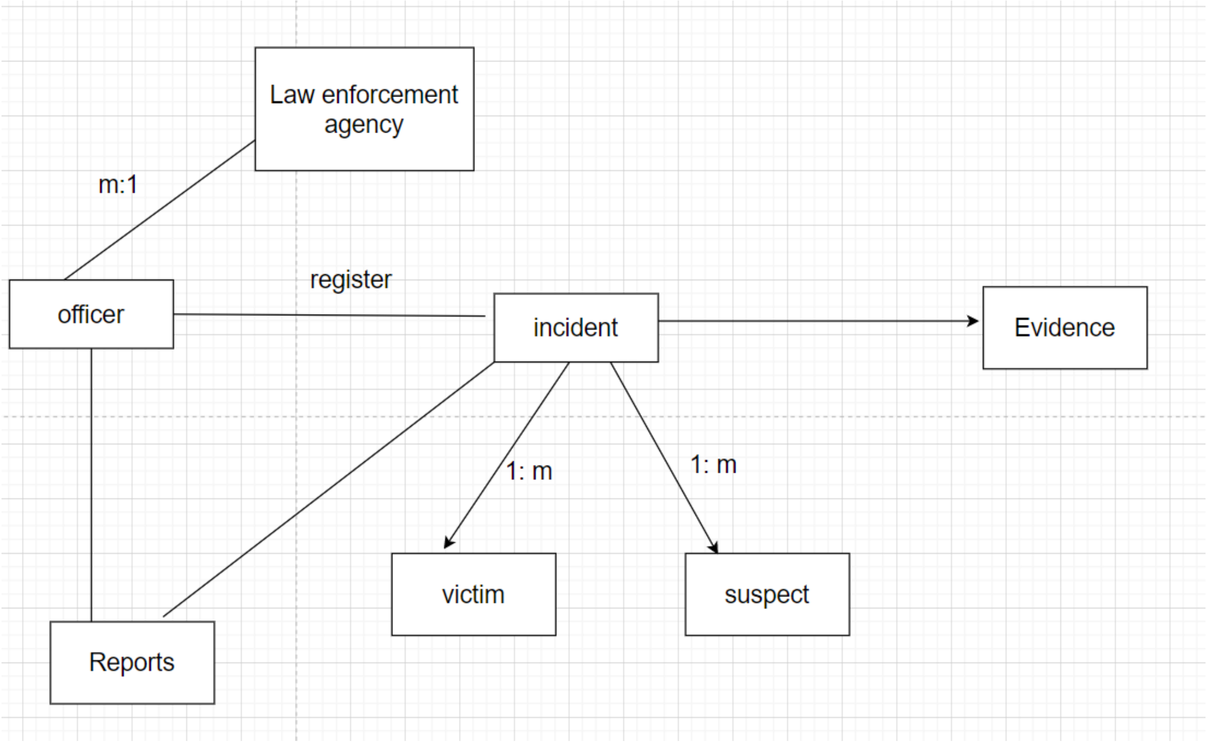
Crime Analysis and Reporting (C A R S)



Instructions

- Project submissions should be done through the partcipants' Github repository and the link should be shared with trainers and Hexavarsity.
- Each section builds upon the previous one, and by the end, you will have a comprehensive Crime Analysis and reporting system implemented with a strong focus on SQL, control flow statements, loops, arrays, collections, exception handling, database interaction and Unit Testing.
- Follow object-oriented principles throughout the project. Use classes and objects to model real-world entities, encapsulate data and behavior, and ensure code reusability.
- Throw user defined exceptions from corresponding methods and handled.

- The following Directory structure is to be followed in the application.
    - **entity**
        - Create entity classes in this package. All entity class should not have any business logic.
    - **dao**
        - Create Service Provider interface to showcase functionalities.
        - Create the implementation class for the above interface with db interaction.
    - **exception**
        - Create user defined exceptions in this package and handle exceptions wheneve needed.
    - **util**
        - Create a DBPropertyUtil class with a static function which takes property file name as parameter and returns connection string.
        - Create a DBConnUtil class which holds static method which takes connection string as parameter file and returns connection object(Use method defined in DBPropertyUtil class to get the connection String).
    - **main**
        - Create a class MainModule and demonstrate the functionalities in a menu driven application.

Key Functionalities:

The primary objective of this project is to develop a comprehensive **Crime Analysis and Reporting System (CARS)** that addresses the above-mentioned challenges and provides law enforcement agencies with a robust, user-friendly, and secure platform for crime data management and reporting.

1. **Schema design:**

Entities:

1. **Incidents:**
    - IncidentID (Primary Key)

- IncidentType (e.g., Robbery, Homicide, Theft)
- IncidentDate
- Location (Geospatial Data: Latitude and Longitude)
- Description
- Status (e.g., Open, Closed, Under Investigation)
- VictimID (Foreign Key, linking to Victims)
- SuspectId(Foreign Key, Linking to Suspect)

2. **Victims:**
    - VictimID (Primary Key)
    - FirstName
    - LastName
    - DateOfBirth
    - Gender
    - Contact Information (e.g., Address, Phone Number)

3. **Suspects:**
    - SuspectID (Primary Key)
    - FirstName
    - LastName
    - DateOfBirth
    - Gender
    - Contact Information

4. **Law Enforcement Agencies:**
    - AgencyID (Primary Key)
    - AgencyName
    - Jurisdiction
    - Contact Information
    - Officer(s) (Link to Officers within the agency)

5. **Officers:**

- OfficerID (Primary Key)
- FirstName
- LastName
- BadgeNumber
- Rank
- Contact Information
- AgencyID (Foreign Key, linking to Law Enforcement Agencies)

6. **Evidence:**
   - EvidenceID (Primary Key)
   - Description
   - Location Found
   - IncidentID (Foreign Key, linking to Incidents)

7. **Reports:**
   - ReportID (Primary Key)
   - IncidentID (Foreign Key, linking to Incidents)
   - ReportingOfficer (Foreign Key, linking to Officers)
   - ReportDate
   - ReportDetails
   - Status (e.g., Draft, Finalized)

Relationships:

- An Incident can have multiple Victims and Suspects.
- An Incident is associated with one Law Enforcement Agency.
- An Officer works for a single Law Enforcement Agency.
- Evidence can be linked to an Incident.
- Reports are generated for Incidents by ReportingOfficers.