#### ****Purpose of Testing****

* Validate core functionalities like creating incidents, updating status, and managing cases.
* Ensure correct database interaction (CRUD operations).
* Verify exception handling for edge cases (e.g., missing incident ID).
* Maintain system reliability as the project evolves.

1. **Testing Strategy**

* **Unit Testing**: Focused on individual methods in the DAO implementation (CrimeAnalysisServiceImpl).
* **Test Class**: TestCrimeAnalysisService includes test methods for each critical feature.

#### ****Test Cases Covered****

* **test\_create\_incident()**  
  Verifies a new incident is inserted correctly with victims and suspects.
* **test\_update\_incident\_status()** Checks if incident status updates work and raises exceptions when IDs are invalid.
* **test\_get\_incidents\_in\_date\_range()**Ensures the system retrieves all incidents between two dates.
* **test\_search\_incidents()**Confirms filtering by incident type is accurate.
* **test\_generate\_incident\_report()**Tests if a dictionary report is generated with all relevant data.
* **test\_create\_case()**Validates whether a case is created with one or more linked incidents.
* **test\_get\_case\_details()**Ensures full case data (description + linked incidents) is fetched.
* **test\_update\_case\_details()**Tests case description update functionality.
* **test\_get\_all\_cases()**Verifies whether all registered cases are returned as a list of objects.

#### ****Exception Handling in Tests****

* Custom exceptions like IncidentNumberNotFoundException are tested for correct triggering.
* Tests gracefully fail with meaningful messages if unexpected issues arise.

#### ****How to Run the Tests****

**python -m unittest test.test\_crime\_analysis**

#### ****Outcome****

* Confirms system robustness
* Increases confidence in database logic
* Helps catch regressions when code changes