GAUTAM SHARMA

DAO

CASE.py

```
from Case_Study.UTIL.DB_CONNECTION import DBConnection


class Cases(DBConnection):
    def __init__(self, case_id=None, description=None, case_date=None,
status=None):
        self.case_id = case_id
        self.description = description
        self.case_date = case_date
        self.status = status

    def create_table(self):
        create_query = '''
            create table if not exists Cases(
            case_id int primary key,
            description varchar(150),
            case_date date,
            status varchar(30)
            )'''
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(create_query)
        print("Case table created successfully")
```

CRIMEANALYSIS_SERVICEIMPL.PY

```
from Case_Study.UTIL.DB_CONNECTION import DBConnection
from Case_Study.ENTITY.ICRIMEANALYSIS_SERVICE import
I_crime_analysis_service
from Case_Study.DAO.INCIDENTS import Incidents
from Case_Study.DAO.REPORTS import Reports
from Case_Study.DAO.CASE import Cases


class crime_analysis_service_impl(Incidents, Reports, Cases, DBConnection,
I_crime_analysis_service):
    def __init__(self):
        super(Incidents, self).__init__()

    def createIncident(self):
        incident = Incidents()
        incident.insert_into()

    def updateIncidentStatus(self):
        incident = Incidents()
        incident.update_table()

    def getIncidentsInDateRange(self):
        start_date = input("Enter the start date(yyyy-mm-dd): ")
        end_date = input("Enter the input date(yyyy-mm-dd): ")
        res = [incident for incident in Incidents.incidents if start_date
<= incident.incident_date <= end_date]
        for i in res:
            print(i)
    def searchIncidents(self):
        self.incident_id = int(input('Enter the incident id to search the
incident details: '))
        search_query = f'select * from Incidents where incident_id =
{self.incident_id}'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(search_query)
        data = stmt.fetchall()
        for i in data:
            print(i)
        print("Search successfully")

    def generateIncidentReport(self):
        self.incident_id = int(input("Enter the incident id to generate a
report: "))
        report_query = f'select * from Reports where incident_id =
{self.incident_id}'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(report_query)
        data = stmt.fetchall()
        for i in data:
            print(i)
        print("Reports generated successfully")

    def createCase(self):
        self.case_id = int(input("Enter the case id: "))
        self.description = input("Enter the description: ")
        self.case_date = input("Enter the case date: ")
        self.status = input("Enter the status: ")

        query = 'insert into Cases(case_id, description, case_date, status)
values(%s,%s,%s,%s)'
        data = [(self.case_id, self.description, self.case_date,
self.status)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.executemany(query, data)
        DBConnection.connection.commit()
        print("Created case successfully")

    def getCaseDetails(self):
        self.case_id = int(input("Enter the case Id to get details: "))
        get_query = f'select * from Cases where case_id={self.case_id}'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(get_query)
        data = stmt.fetchall()
        for i in data:
            print(i)
        print("Case details displayed successfully")

    def updateCaseDetails(self):
        self.case_id = int(input("Enter the case Id to update details: "))
        self.description = input("Enter the description: ")
        self.case_date = input("Enter the case date: ")
        self.status = input("Enter the status: ")
        update_query = f'update Cases set description=%s, case_date=%s,
status=%s where case_id=%s'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(update_query)
        print("Case updated successfully")

    def getAllCases(self):
        get_query = f'select * from Cases'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(get_query)
        data = stmt.fetchall()
        for i in data:
            print(i)



# obj = crime_analysis_service_impl()
# obj.generateIncidentReport()
```

EVIDENCE.PY

```
from Case_Study.UTIL.DB_CONNECTION import DBConnection


class Evidence(DBConnection):
    def __init__(self, evidence_id=None, description=None, location=None,
incident_id=None):
        self.evidence_id = evidence_id
        self.description = description
        self.location = location
        self.incident_id = incident_id

    def create_table(self):
        create_query = '''
        create table if not exists Evidence(
        evidence_id int primary key,
        description varchar(150),
        location varchar(50),
        incident_id int
        )'''
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(create_query)
        print("Evidence table created successfully")

    def insert_into(self):
        self.evidence_id = int(input("Enter the evidence id: "))
        self.description = input("Enter the description: ")
        self.location = input("Enter the location: ")
        self.incident_id = input("Enter the incident id: ")

        insert_query = 'insert into Evidence(evidence_id, description,
location, incident_id) values(%s,%s,%s,%s)'
        data = [(self.evidence_id, self.description, self.location,
self.incident_id)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.executemany(insert_query, data)
        DBConnection.connection.commit()
        print("Values inserted successfully")
```

```python
    def update_table(self):
        self.evidence_id = int(input("Enter the evidence id: "))
        self.description = input("Enter the description: ")
        self.location = input("Enter the location: ")
        self.incident_id = input("Enter the incident id: ")

        update_query = 'update Evidence set description=%s, location=%s, 
incident_id=%s where evidence_id=%s'
        data = [(self.description, self.location, self.incident_id, 
self.evidence_id)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(update_query, data)
        DBConnection.connection.commit()
        print("Values updated successfully")

    def delete_table(self):
        self.evidence_id = int(input("Enter the evidence id to delete 
values: "))
        delete_query = f'delete from Evidence where 
evidence_id={self.evidence_id}'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(delete_query)
        DBConnection.connection.commit()
        print("Values deleted successfully")

    def select_table(self):
        select_query = 'select * from Evidence'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(select_query)
        data = stmt.fetchall()
        for i in data:
            print(i)
        print("Values displayed successfully")
```

INCIDENTS.PY

```python
from Case_Study.UTIL.DB_CONNECTION import DBConnection
from Case_Study.EXCEPTION.INCIDENTNUMBERNOTFOUND import 
IncidentNumberNotFoundException

class Incidents(DBConnection):
    incidents = []

    def __init__(self, incident_id=None, incident_type=None, 
incident_date=None, location=None, description=None, status=None, 
victim_id=None, suspect_id=None):
        self.incident_id = incident_id
        self.incident_type = incident_type
        self.incident_date = incident_date
        self.location = location
        self.description = description
        self.status = status
        self.victim_id = victim_id
        self.suspect_id = suspect_id

    def create_table(self):
        create_query = '''
```

```
            create table if not exists Incidents(
            incident_id int primary key,
            incident_type varchar(30),
            incident_date date,
            location varchar(30),
            description varchar(100),
            status varchar(30),
            victim_id int,
            suspect_id int
            )
        '''
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(create_query)
        print("Incidents table created successfully")

    def insert_into(self):
        self.incident_id = int(input("Enter the incident id: "))
        self.incident_type = input("Enter the incident type: ")
        self.incident_date = input("Enter the incident date: ")
        self.location = input("Enter the location: ")
        self.description = input("Enter the description: ")
        self.status = input("Enter the status: ")
        self.victim_id = int(input("Enter the victim id: "))
        self.suspect_id = int(input("Enter the suspect id: "))

        insert_query = 'insert into Incidents(incident_id, incident_type,
incident_date, location, description, status, victim_id, suspect_id)
values(%s,%s,%s,%s,%s,%s,%s,%s)'
        data = [(self.incident_id, self.incident_type, self.incident_date,
self.location, self.description, self.status, self.victim_id,
self.suspect_id)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.executemany(insert_query, data)
        DBConnection.connection.commit()
        print("Data inserted successfully")
        return 'Incident created successfully'

    def update_table(self):
        try:
            self.incident_id = int(input("Enter the incident id to update
the values: "))
            self.incident_type = input("Enter the incident type: ")
            self.incident_date = input("Enter the incident date: ")
            self.location = input("Enter the location: ")
            self.description = input("Enter the description: ")
            self.status = input("Enter the status: ")
            self.victim_id = int(input("Enter the victim id: "))
            self.suspect_id = int(input("Enter the suspect id: "))
            if not self.incident_exists(self.incident_id):
                raise IncidentNumberNotFoundException("Incident id not
found")

            update_query = 'update Incidents set incident_type=%s,
incident_date=%s, location=%s, description=%s, status=%s, victim_id=%s,
suspect_id=%s where incident_id=%s'
            data = [(self.incident_type, self.incident_date, self.location,
self.description, self.status, self.victim_id, self.suspect_id,
self.incident_id)]
            DBConnection.getConnection()
            stmt = DBConnection.connection.cursor()
            stmt.executemany(update_query, data)
            DBConnection.connection.commit()
            print("Updated successfully")
            return 'Values updated successfully'

        except IncidentNumberNotFoundException as e:
            print(e)
        except Exception as e:
            print(e)

    def delete_table(self):
        try:
            self.incident_id = int(input("Enter the incident id to delete 
data: "))
            if not self.incident_exists(self.incident_id):
                raise IncidentNumberNotFoundException("Incident id not 
found")
            delete_query = f'delete from Incidents where incident_id = 
{self.incident_id}'
            DBConnection.getConnection()
            stmt = DBConnection.connection.cursor()
            stmt.execute(delete_query)
            DBConnection.connection.commit()
            print("Deleted successfully")

        except IncidentNumberNotFoundException as e:
            print(e)
        except Exception as e:
            print(e)

    def select_table(self):
        select_query = 'select * from Incidents'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(select_query)
        data = stmt.fetchall()
        for i in data:
            Incidents.incidents.append(i)
            print(i)
        Incidents.incidents = [list(i) for i in Incidents.incidents]
        print("Values displayed successfully")

    def incident_exists(self, incident_id):
        try:
            DBConnection.getConnection()
            stmt = DBConnection.connection.cursor()
            select_query = f'SELECT COUNT(*) FROM Incidents WHERE 
incident_id = {incident_id}'
            stmt.execute(select_query)
            result = stmt.fetchone()
            if result and result[0] > 0:
                return True
            else:
                return False

        except Exception as e:
            print(f"Error checking incident existence: {e}")
            return False

    def __str__(self):
        return """f'Incident ID: {self.incident_id}', f'Incident Type:
{self.incident_type}',f'Incident date: {self.incident_date}',
         f'Location: {self.location}', f'Description: {self.description}',
f'Status: {self.status}',
          f'Victim ID: {self.victim_id}', f'Suspect ID: {self.suspect_id}'
"""
```

LAW_ENFORCEMENT_AGENICES.PY

```
from Case_Study.UTIL.DB_CONNECTION import DBConnection


class Law_Enforcement_Agencies(DBConnection):
    def __init__(self, agency_id=None, agency_name=None, jurisdiction=None,
phone_num=None, officer=None):
        self.agency_id = agency_id
        self.agency_name = agency_name
        self.jurisdiction = jurisdiction
        self.phone_num = phone_num
        self.officer = officer

    def create_table(self):
        create_query = '''
            create table Law_Enforcement_Agencies(
            agency_id int primary key,
            agency_name varchar(30),
            jurisdiction varchar(50),
            phone_num varchar(20),
            officer varchar(30)
        )'''
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(create_query)
        print("Law Enforcement Agencies table created successfully")

    def insert_into(self):
        self.agency_id = int(input("Enter the agency id: "))
        self.agency_name = input("Enter the agency name: ")
        self.jurisdiction = input("Enter the jurisdiction: ")
        self.phone_num = input("Enter the phone number: ")
        self.officer = input("Enter the officer: ")

        insert_query = 'insert into Law_Enforcement_Agencies(agency_id,
agency_name, jurisdiction, phone_num, officer) values(%s,%s,%s,%s,%s)'
        data = [(self.agency_id, self.agency_name, self.jurisdiction,
self.phone_num, self.officer)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.executemany(insert_query, data)
        DBConnection.connection.commit()
        print("Values inserted successfully")

    def update_table(self):
        self.agency_id = int(input("Enter the agency id to update the
values: "))
```

```
        self.agency_name = input("Enter the agency name: ")
        self.jurisdiction = input("Enter the jurisdiction: ")
        self.phone_num = input("Enter the phone number: ")
        self.officer = input("Enter the officer: ")

        update_query = 'update Law_Enforcements_Agencies set 
agency_name=%s, jurisdiction=%s, phone_num=%s, officer=%s where 
agency_id=%s'
        data = [(self.agency_name, self.jurisdiction, self.phone_num, 
self.officer, self.agency_id)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(update_query, data)
        DBConnection.connection.commit()
        print("Values updated successfully")

    def delete_table(self):
        self.agency_id = int(input("Enter the agency id to delete values: 
"))
        delete_query = f'delete from Law_Enforcement_Agencies where 
agency_id={self.agency_id}'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(delete_query)
        DBConnection.connection.commit()
        print("Values deleted successfully")

    def select_table(self):
        select_query = 'select * from Law_Enforcement_Agencies'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(select_query)
        data = stmt.fetchall()
        for i in data:
            print(i)
        print("Values displayed successfully")
```

OFFICERS.PY

```
from Case_Study.UTIL.DB_CONNECTION import DBConnection


class Officers(DBConnection):
    def __init__(self, officer_id=None, first_name=None, last_name=None, 
badge_no=None, officer_rank=None, phone_num=None, agency_id=None):
        self.officer_id = officer_id
        self.first_name = first_name
        self.last_name = last_name
        self.badge_no = badge_no
        self.officer_rank = officer_rank
        self.phone_num = phone_num
        self.agency_id = agency_id

    def create_table(self):
        create_query = ''' create table if not exists Officers(
            officer_id int primary key,
            first_name varchar(30),
            last_name varchar(30),
            badge_no varchar(10),
            officer_rank varchar(30),
            phone_num varchar(20),
            agency_id int
        )'''
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(create_query)
        print("Officers table created successfully")

    def insert_into(self):
        self.officer_id = int(input("Enter the officer id: "))
        self.first_name = input("Enter the first name: ")
        self.last_name = input("Enter the last name: ")
        self.badge_no = input("Enter the badge number: ")
        self.officer_rank = input("Enter the rank: ")
        self.phone_num = input("Enter the phone number: ")
        self.agency_id = input("Enter the agency id: ")

        insert_query = 'insert into Officers(officer_id, first_name,
last_name, badge_no, officer_rank, phone_num, agency_id)
values(%s,%s,%s,%s,%s,%s,%s)'
        data = [(self.officer_id, self.first_name, self.last_name,
self.badge_no, self.officer_rank, self.phone_num, self.agency_id)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.executemany(insert_query, data)
        DBConnection.connection.commit()
        print("Values inserted successfully")

    def update_table(self):
        self.officer_id = int(input("Enter the officer id to update values:
"))
        self.first_name = input("Enter the first name: ")
        self.last_name = input("Enter the last name: ")
        self.badge_no = input("Enter the badge number: ")
        self.officer_rank = input("Enter the rank: ")
        self.phone_num = input("Enter the phone number: ")
        self.agency_id = input("Enter the agency id: ")

        update_query = 'update Officers set first_name=%s, last_name=%s,
badge_no=%s, officer_rank=%s, phone_num=%s, agency_id=%s where
officer_id=%s'
        data = [(self.first_name, self.last_name, self.badge_no,
self.officer_rank, self.phone_num, self.agency_id, self.officer_id)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(update_query, data)
        DBConnection.connection.commit()
        print("Values updated successfully")

    def delete_table(self):
        self.officer_id = int(input("Enter the officer id to delete values:
"))
        delete_query = f'delete from Officers where
officer_id={self.officer_id}'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(delete_query)
        DBConnection.connection.commit()
        print("Values deleted successfully")
    def select_table(self):
        select_query = 'select * from Officers'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(select_query)
        data = stmt.fetchall()
        for i in data:
            print(i)
        print("Values displayed successfully")
```

REPORTS.PY

```
from Case_Study.UTIL.DB_CONNECTION import DBConnection


class Reports(DBConnection):
    def __init__(self, report_id=None, incident_id=None,
reporting_officer=None, report_date=None, report_details=None,
status=None):
        self.report_id = report_id
        self.incident_id = incident_id
        self.reporting_officer = reporting_officer
        self.report_date = report_date
        self.report_details = report_details
        self.status = status

    def create_table(self):
        create_query = '''
        create table if not exists Reports(
        report_id int primary key,
        incident_id int,
        reporting_officer varchar(30),
        report_date date,
        report_details varchar(150),
        status varchar(20)
        )'''
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(create_query)
        print("Reports table successfully created")

    def insert_into(self):
        self.report_id = int(input("Enter the report id: "))
        self.incident_id = input("Enter the incident id: ")
        self.reporting_officer = input("Enter the reporting officer: ")
        self.report_date = input("Enter the report date: ")
        self.report_details = input("Enter the report details: ")
        self.status = input("Enter the status: ")

        insert_query = 'insert into Reports(report_id, incident_id,
reporting_officer, report_date, report_details, status)
values(%s,%s,%s,%s,%s,%s)'
        data = [(self.report_id, self.incident_id, self.reporting_officer,
self.report_date, self.report_details, self.status)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.executemany(insert_query, data)
        DBConnection.connection.commit()
        print("Values inserted successfully")
    def update_table(self):
        self.report_id = int(input("Enter the report id: "))
        self.incident_id = input("Enter the incident id: ")
        self.reporting_officer = input("Enter the reporting officer: ")
        self.report_date = input("Enter the report date: ")
        self.report_details = input("Enter the report details: ")
        self.status = input("Enter the status: ")

        update_query = 'update Reports set incident_id=%s,
reporting_officer=%s, report_date=%s, report_details=%s, status=%s where
report_id=%s'
        data = [(self.incident_id, self.reporting_officer,
self.report_date, self.report_details, self.status, self.report_id)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(update_query, data)
        DBConnection.connection.commit()
        print("Values updated successfully")

    def delete_table(self):
        self.report_id = int(input("Enter the report id to delete values:
"))
        delete_query = f'delete from Reports where
report_id={self.report_id}'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(delete_query)
        DBConnection.connection.commit()
        print("Values deleted successfully")

    def select_table(self):
        select_query = 'select * from Reports'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(select_query)
        data = stmt.fetchall()
        for i in data:
            print(i)
        print("Values displayed successfully")
```

SUSPECTS.PY

```
from Case_Study.UTIL.DB_CONNECTION import DBConnection


class Suspects(DBConnection):
    def __init__(self, suspect_id=None, first_name=None, last_name=None,
dob=None, gender=None, address=None, phone_num=None):
        self.suspect_id = suspect_id
        self.first_name = first_name
        self.last_name = last_name
        self.dob = dob
        self.gender = gender
        self.address = address
        self.phone_num = phone_num

    def create_table(self):
        create_query = '''
            create table if not exists Suspects(
            suspect_id int primary key,
            first_name varchar(30),
            last_name varchar(30),
            dob date,
            gender char,
            address varchar(30),
            phone_num varchar(20))
            '''
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(create_query)
        print("Suspects table created successfully")

    def insert_into(self):
        self.suspect_id = int(input("Enter the suspect id: "))
        self.first_name = input("Enter the first name: ")
        self.last_name = input("Enter the last name: ")
        self.dob = input("Enter the date of birth: ")
        self.gender = input("Enter the gender: ")
        self.address = input("Enter the address: ")
        self.phone_num = input("Enter the phone number: ")

        insert_query = 'insert into Suspects(suspect_id, first_name,
last_name, dob, gender, address, phone_num) values(%s,%s,%s,%s,%s,%s,%s)'
        data = [(self.suspect_id, self.first_name, self.last_name,
self.dob, self.gender, self.address, self.phone_num)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.executemany(insert_query, data)
        DBConnection.connection.commit()
        print("Values inserted successfully")

    def update_table(self):
        self.suspect_id = int(input("Enter the suspect id to update the
values: "))
        self.first_name = input("Enter the first name: ")
        self.last_name = input("Enter the last name: ")
        self.dob = input("Enter the date of birth: ")
        self.gender = input("Enter the gender: ")
        self.address = input("Enter the address: ")
        self.phone_num = input("Enter the phone number: ")

        update_query = 'update Suspects set first_name=%s, last_name=%s,
dob=%s, gender=%s, address=%s, phone_num=%s where suspect_id=%s'
        data = [(self.first_name, self.last_name, self.dob, self.gender,
self.address, self.phone_num, self.suspect_id)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(update_query, data)
        DBConnection.connection.commit()
        print("Values updated successfully")

    def delete_table(self):
        self.suspect_id = int(input("Enter the suspect id to delete values:
"))
        delete_query = f'delete from Suspects where
suspect_id={self.suspect_id}'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(delete_query)
        DBConnection.connection.commit()
        print("Values deleted successfully")
    def select_table(self):
        select_query = 'select * from Suspects'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(select_query)
        data = stmt.fetchall()
        for i in data:
            print(i)
        print("Values displayed successfully")
```

VICTIMS.PY

```
from Case_Study.UTIL.DB_CONNECTION import DBConnection


class Victims(DBConnection):
    def __init__(self, victim_id=None, first_name=None, last_name=None,
dob=None, gender=None, address=None, phone_num=None):
        self.victim_id = victim_id
        self.first_name = first_name
        self.last_name = last_name
        self.dob = dob
        self.gender = gender
        self.address = address
        self.phone_num = phone_num

    def create_table(self):
        create_query = '''
            create table if not exists Victims(
            victim_id int primary key,
            first_name varchar(30),
            last_name varchar(30),
            dob date,
            gender char,
            address varchar(30),
            phone_num varchar(20))
            '''
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(create_query)
        print("Victims table created successfully")

    def insert_into(self):
        self.victim_id = int(input("Enter the victim id: "))
        self.first_name = input("Enter the first name: ")
        self.last_name = input("Enter the last name: ")
        self.dob = input("Enter the date of birth: ")
        self.gender = input("Enter the gender: ")
        self.address = input("Enter the address: ")
        self.phone_num = input("Enter the phone number: ")

        insert_query = 'insert into Victims(victim_id, first_name,
last_name, dob, gender, address, phone_num) values(%s,%s,%s,%s,%s,%s,%s)'
        data = [(self.victim_id, self.first_name, self.last_name, self.dob,
self.gender, self.address, self.phone_num)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.executemany(insert_query, data)
        DBConnection.connection.commit()
        print("Values inserted successfully")
    def update_table(self):
        self.victim_id = int(input("Enter the victim id to update the
values: "))
        self.first_name = input("Enter the first name: ")
        self.last_name = input("Enter the last name: ")
        self.dob = input("Enter the date of birth: ")
        self.gender = input("Enter the gender: ")
        self.address = input("Enter the address: ")
        self.phone_num = input("Enter the phone number: ")

        update_query = 'update Victims set first_name=%s, last_name=%s,
dob=%s, gender=%s, address=%s, phone_num=%s where victim_id=%s'
        data = [(self.first_name, self.last_name, self.dob, self.gender,
self.address, self.phone_num, self.victim_id)]
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(update_query, data)
        DBConnection.connection.commit()
        print("Values updated successfully")

    def delete_table(self):
        self.victim_id = int(input("Enter the victim id to delete values:
"))
        delete_query = f'delete from Victims where
victim_id={self.victim_id}'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(delete_query)
        DBConnection.connection.commit()
        print("Values deleted successfully")

    def select_table(self):
        select_query = 'select * from Victims'
        DBConnection.getConnection()
        stmt = DBConnection.connection.cursor()
        stmt.execute(select_query)
        data = stmt.fetchall()
        for i in data:
            print(i)
        print("Values displayed successfully")
```

ENTITY

ICRIMEANALYSIS_SERVICE.PY

```
from abc import ABC, abstractmethod


class I_crime_analysis_service:

    @abstractmethod
    def createIncident(self):
        pass

    @abstractmethod
    def updateIncidentStatus(self):
        pass

    @abstractmethod
    def getIncidentsInDateRange(self):
        pass

    @abstractmethod
    def searchIncidents(self):
        pass

    @abstractmethod
    def generateIncidentReport(self):
        pass

    @abstractmethod
    def createCase(self):
        pass

    @abstractmethod
    def getCaseDetails(self):
        pass

    @abstractmethod
    def updateCaseDetails(self):
        pass

    @abstractmethod
    def getAllCases(self):
        pass
```

EXCEPTION

INCIDENTNUMBERNOTFOUND.PY

```
class IncidentNumberNotFoundException(Exception):
    def __init__(self, msg="Incident id not found"):
        self.msg = msg
        super().__init__(msg)
```

PYUNIT

TESTCASES.PY

```
import unittest
from Case_Study.DAO.INCIDENTS import Incidents


class MyTestCase(unittest.TestCase):
    def setUp(self):
        self.incident = Incidents()

    # testing whether an incident is created or not
    def test_incident(self):
        print("Create a new incident with incident id =5")
        result = self.incident.insert_into()
        self.assertEqual('Incident created successfully', result)

    # testing whether incident status updated or not
    def test_update(self):
        print("Updating the status of incident. Set status = Investigation
")
        result = self.incident.update_table()
        self.assertEqual('Values updated successfully', result)
```

GAUTAM SHARMA

```
if __name__ == '__main__':
    unittest.main()
```

UTIL

DB_CONNECTION.PY

```
import mysql.connector as sql
from mysql.connector import Error
from Case_Study.UTIL.PROPERTY_UTIL import propertyUtil

class DBConnection:
    connection = None

    @staticmethod
    def getConnection():
        if DBConnection.connection is None:
            try:
                connection_string = propertyUtil.getPropertyString()
                DBConnection.connection = sql.connect(**connection_string)

                if DBConnection.connection.is_connected():
                    print("Database connected successfully")

            except Error as e:
                print(f'Error : {e}')

        return DBConnection.connection
```

PROPERTY_UTIL.PY

```
class propertyUtil:

    @staticmethod
    def getPropertyString():
        connection_string = {
            'host': 'localhost',
            'database': 'crime_reporting_system',
            'user': 'root',
            'password': 'root'
        }

        return connection_string
```

MAIN_MODULE

MAIN

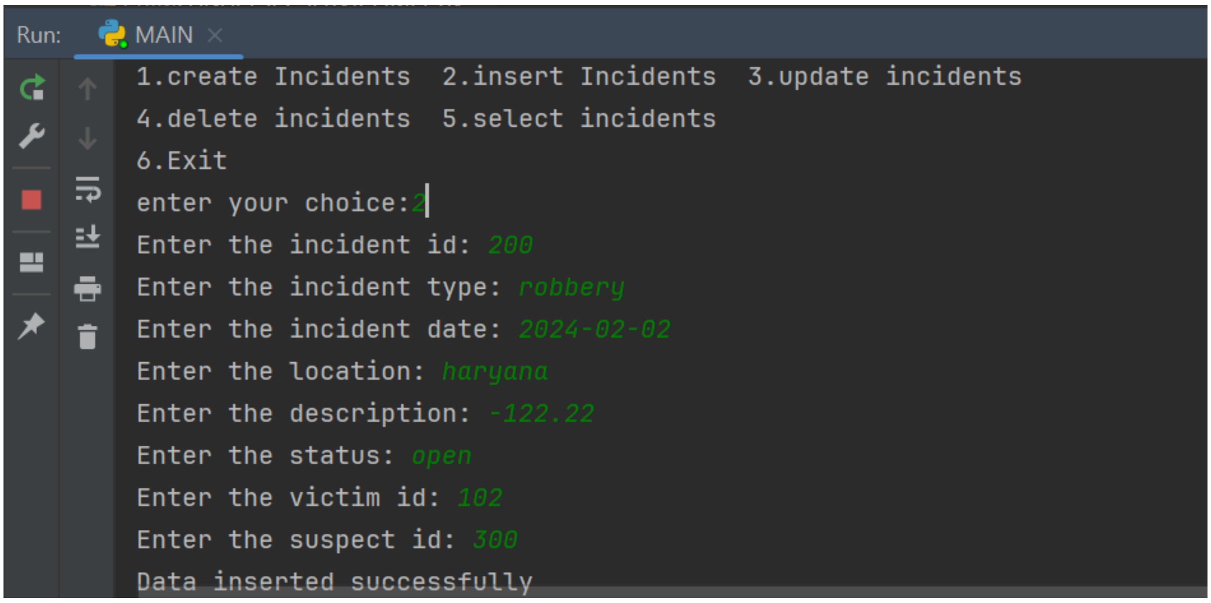
```
from Case_Study.UTIL.DB_CONNECTION import DBConnection
from Case_Study.DAO.INCIDENTS import Incidents
from Case_Study.DAO.VICTIMS import Victims
from Case_Study.DAO.SUSPECTS import Suspects
from Case_Study.DAO.LAW_ENFORCEMENT_AGENCIES import
Law_Enforcement_Agencies
from Case_Study.DAO.OFFICERS import Officers
from Case_Study.DAO.EVIDENCE import Evidence
from Case_Study.DAO.REPORTS import Reports
```

```python
from Case_Study.DAO.CASE import Cases
from Case_Study.DAO.CRIMEANALYSIS_SERVICEIMPL import
crime_analysis_service_impl

try:
    connObj = DBConnection()
    con = connObj.getConnection()

    while True:
        incidentObj = Incidents()
        victimObj = Victims()
        suspectObj = Suspects()
        lawObj = Law_Enforcement_Agencies()
        officerObj = Officers()
        evidenceObj = Evidence()
        reportObj = Reports()
        serviceImplementObj = crime_analysis_service_impl()

        print("Select table to use functionalities")
        print("1.Incidents\n2.Victims\n3.Suspects\n4.Law Enforcement
Agencies\n5.Officers\n6.Evidence\n7.Reports\n8.crime analysis service
impl\n9.exit")
        ch = int(input("enter your choice:"))

        if ch == 1:
            while True:
                print("1.create Incidents\t2.insert Incidents\t3.update
incidents\n4.delete incidents\t5.select incidents\n6.Exit")
                choice = int(input("enter your choice:"))
                if choice == 1:
                    incidentObj.create_table()
                elif choice == 2:
                    incidentObj.insert_into()
                elif choice == 3:
                    incidentObj.update_table()
                elif choice == 4:
                    incidentObj.delete_table()
                elif choice == 5:
                    incidentObj.select_table()
                elif choice == 6:
                    print("exited successfully")
                    break
                else:
                    print("Wrong choice")

        elif ch == 2:
            while True:
                print("1.create victims\t2.insert victims\t3.update
victims\n4.delete victims\t5.select victims\n6.Exit")
                choice = int(input("enter your choice:"))
                if choice == 1:
                    victimObj.create_table()
                elif choice == 2:
                    victimObj.insert_into()
                elif choice == 3:
                    victimObj.update_table()
                elif choice == 4:
                    victimObj.delete_table()
                elif choice == 5:
                    victimObj.select_table()
                elif choice == 6:
                    print("Exited successfully")
                    break
                else:
                    print("Wrong choice")

        elif ch == 3:
            while True:
                print("1.create suspects\t2.insert suspects\t3.update
suspects\n4.delete suspects\t5.select suspects\n6.Exit")
                choice = int(input("enter your choice:"))
                if choice == 1:
                    suspectObj.create_table()
                elif choice == 2:
                    suspectObj.insert_into()
                elif choice == 3:
                    suspectObj.update_table()
                elif choice == 4:
                    suspectObj.delete_table()
                elif choice == 5:
                    (suspectObj.select_table())
                elif choice == 6:
                    print("Exited successfully")
                    break
                else:
                    print("Wrong choice")

        elif ch == 4:
            while True:
                print("1.create law agencies\t2.insert law
agencies\t3.update law agencies\n4.delete law agencies\t5.select law
agencies\n6.Exit")
                choice = int(input("enter your choice:"))
                if choice == 1:
                    lawObj.create_table()
                elif choice == 2:
                    lawObj.insert_into()
                elif choice == 3:
                    lawObj.update_table()
                elif choice == 4:
                    lawObj.delete_table()
                elif choice == 5:
                    lawObj.select_table()
                elif choice == 6:
                    print("Exited successfully")
                    break
                else:
                    print("Wrong choice")

        elif ch == 5:
            while True:
                print("1.create officers\t2.insert officers\t3.update
officers\n4.delete officers\t5.select officers\n6.Exit")
                choice = int(input("enter your choice"))
                if choice == 1:
                    officerObj.create_table()
                elif choice == 2:
                    officerObj.insert_into()
                elif choice == 3:
                    officerObj.update_table()
                elif choice == 4:
                    officerObj.delete_table()
                elif choice == 5:
                    officerObj.select_table()
                elif choice == 6:
                    print("Exited successfully")
                    break
                else:
                    print("Wrong choice")

        elif ch == 6:
            while True:
                print("1.create evidence\t2.insert evidence\t3.update
evidence\n4.delete evidence\t5.select evidence\n6.Exit")
                choice = int(input("enter your choice"))
                if choice == 1:
                    evidenceObj.create_table()
                elif choice == 2:
                    evidenceObj.insert_into()
                elif choice == 3:
                    evidenceObj.update_table()
                elif choice == 4:
                    evidenceObj.delete_table()
                elif choice == 5:
                    evidenceObj.select_table()
                elif choice == 6:
                    print("Exited successfully")
                    break
                else:
                    print("Wrong choice")
        elif ch == 7:
            while True:
                print("1.create reports\t2.insert reports\t3.update
reports\n4.delete reports\t5.select reports\n6.Exit")
                choice = int(input("enter your choice"))
                if choice == 1:
                    reportObj.create_table()
                elif choice == 2:
                    reportObj.insert_into()
                elif choice == 3:
                    reportObj.update_table()
                elif choice == 4:
                    reportObj.delete_table()
                elif choice == 5:
                    reportObj.select_table()
                elif choice == 6:
                    print("Exited successfully")
                    break
                else:
                    print("Wrong choice")

        elif ch == 8:
            while True:
                print("1.create Incident\t2.update incident status\t3.get
incidents in date range officers\n4.search incidents\t5.generate incident
report\n6.create case\t7.get case details\t8.update case details\t9.get all
cases\t10.exit")
                choice = int(input("enter your choice"))
                if choice == 1:
                    serviceImplementObj.createIncident()
                elif choice == 2:
                    serviceImplementObj.updateIncidentStatus()
                elif choice == 3:
                    serviceImplementObj.getIncidentsInDateRange()
                elif choice == 4:
                    serviceImplementObj.searchIncidents()
                elif choice == 5:
                    serviceImplementObj.generateIncidentReport()
                elif choice == 6:
                    serviceImplementObj.createCase()
                elif choice == 7:
                    serviceImplementObj.getCaseDetails()
                elif choice == 8:
                    serviceImplementObj.updateCaseDetails()
                elif choice == 9:
                    serviceImplementObj.getAllCases()
                elif choice == 10:
                    print("Exited successfully")
                    break
                else:
                    print("Wrong choice")

        elif ch == 9:
            print("Exited successfully")
            break

        else:
            print("Wrong choice")

except Exception as e:
    print(f"Unhandled error: {e}")

finally:
    DBConnection.connection.close()
    print("Database connection closed")
```

GAUTAM SHARMA

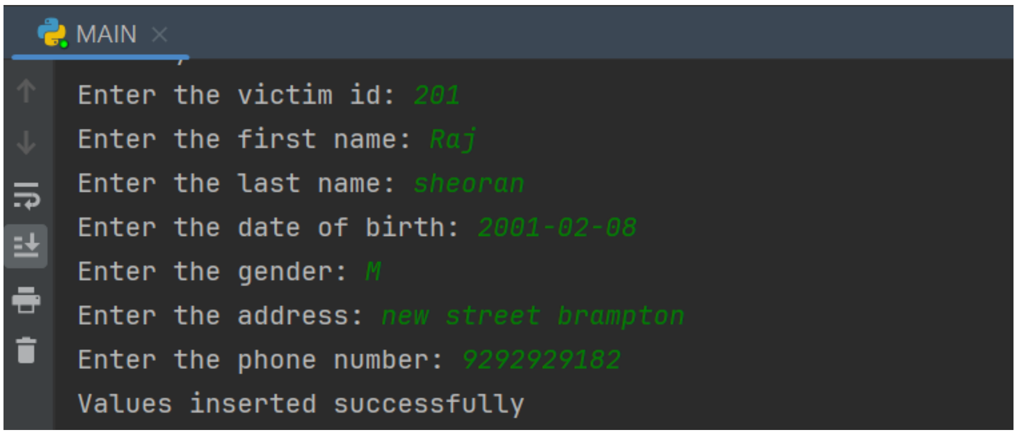
OUTPUTS-

1) Creating and inserting in incident table.



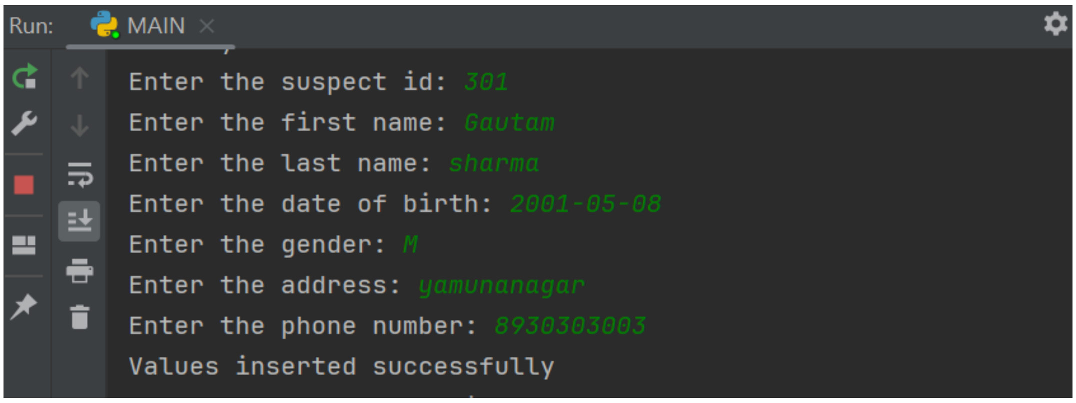
```
1.create Incidents  2.insert Incidents  3.update incidents
4.delete incidents  5.select incidents
6.Exit
enter your choice:2
Enter the incident id: 200
Enter the incident type: robbery
Enter the incident date: 2024-02-02
Enter the location: haryana
Enter the description: -122.22
Enter the status: open
Enter the victim id: 102
Enter the suspect id: 300
Data inserted successfully
```

2) Creating and inserting in victim table.



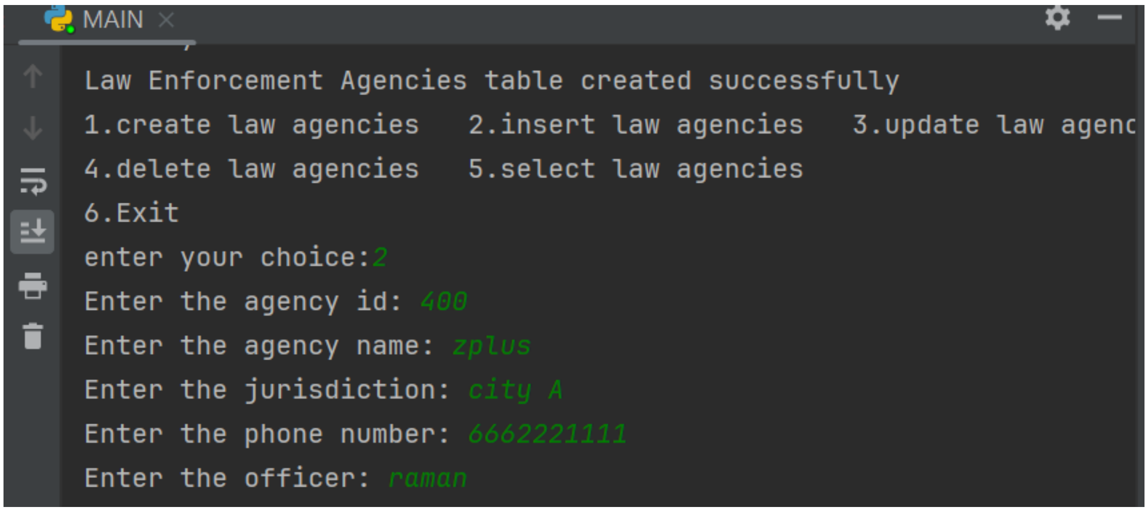
```
Enter the victim id: 201
Enter the first name: Raj
Enter the last name: sheoran
Enter the date of birth: 2001-02-08
Enter the gender: M
Enter the address: new street brampton
Enter the phone number: 9292929182
Values inserted successfully
```

3) Inserting values in suspect table.



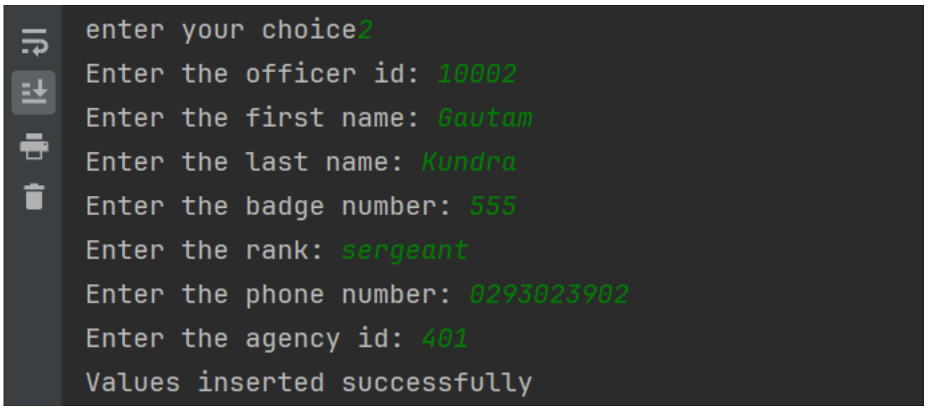
```
Enter the suspect id: 301
Enter the first name: Gautam
Enter the last name: sharma
Enter the date of birth: 2001-05-08
Enter the gender: M
Enter the address: yamunanagar
Enter the phone number: 8930303003
Values inserted successfully
```

4) Creating and inserting values for law enforcement agencies table.



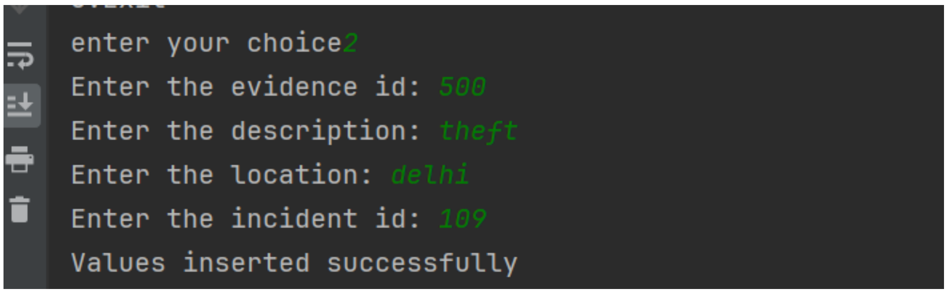
```
Law Enforcement Agencies table created successfully
1.create law agencies   2.insert law agencies   3.update law agenc
4.delete law agencies   5.select law agencies
6.Exit
enter your choice:2
Enter the agency id: 400
Enter the agency name: zplus
Enter the jurisdiction: city A
Enter the phone number: 6662221111
Enter the officer: raman
```

5) Inserting values for officer table.



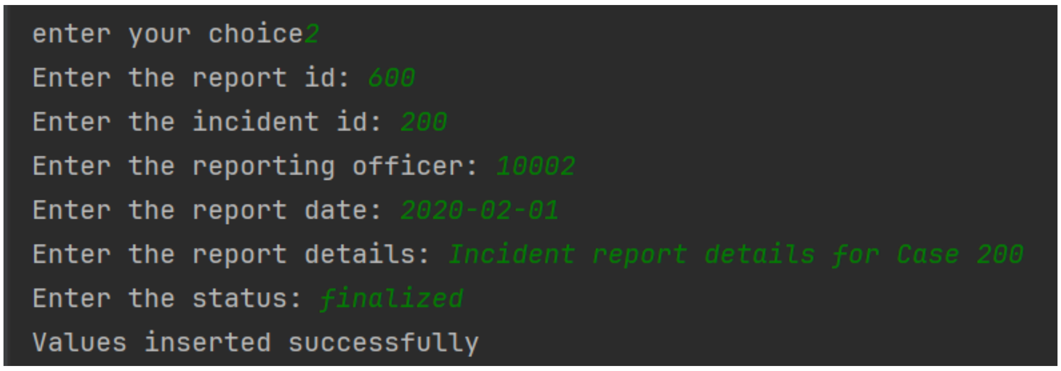
```
enter your choice2
Enter the officer id: 10002
Enter the first name: Gautam
Enter the last name: Kundra
Enter the badge number: 555
Enter the rank: sergeant
Enter the phone number: 0293023902
Enter the agency id: 401
Values inserted successfully
```

6) Inserting values for evidence table.



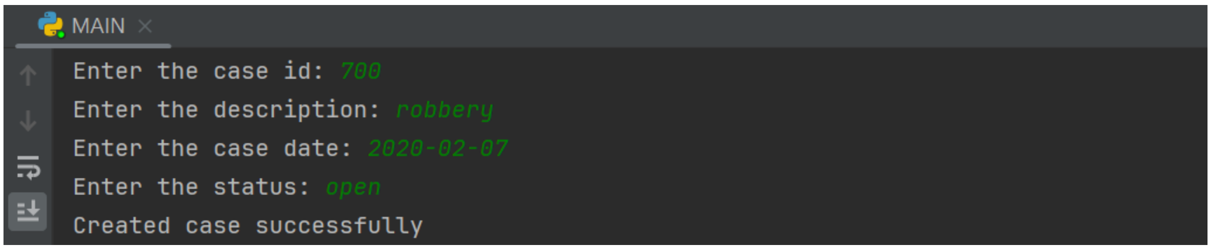
```
enter your choice2
Enter the evidence id: 500
Enter the description: theft
Enter the location: delhi
Enter the incident id: 109
Values inserted successfully
```

7) Inserting values for report table.



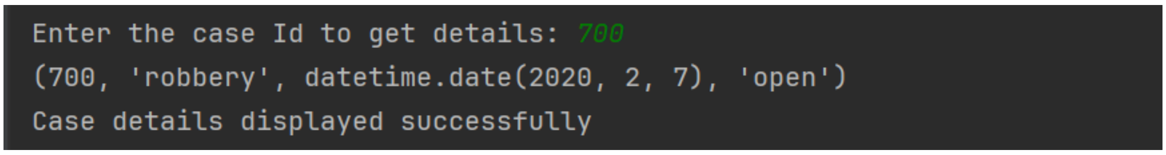
```
enter your choice2
Enter the report id: 600
Enter the incident id: 200
Enter the reporting officer: 10002
Enter the report date: 2020-02-01
Enter the report details: Incident report details for Case 200
Enter the status: finalized
Values inserted successfully
```

8) Inserting values for case table.



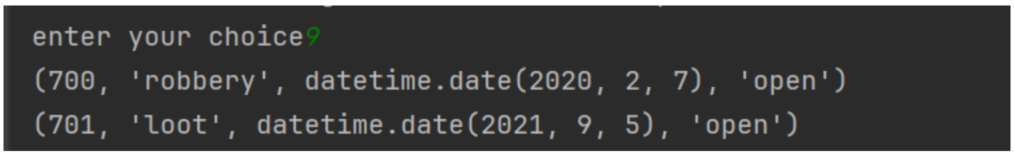
```
Enter the case id: 700
Enter the description: robbery
Enter the case date: 2020-02-07
Enter the status: open
Created case successfully
```

9) Getting case details.



```
Enter the case Id to get details: 700
(700, 'robbery', datetime.date(2020, 2, 7), 'open')
Case details displayed successfully
```

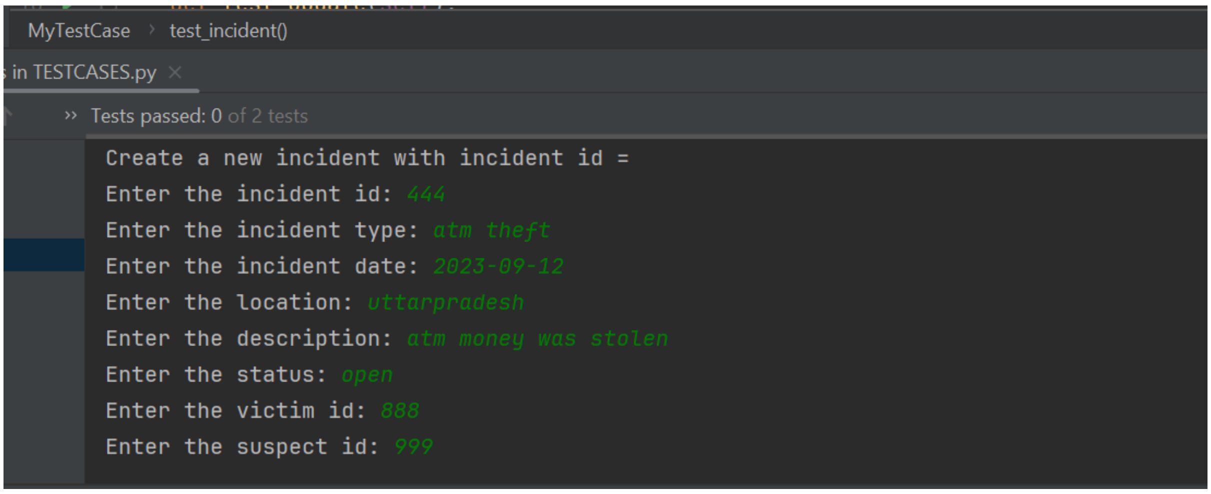
10) Getting all case details.



```
enter your choice9
(700, 'robbery', datetime.date(2020, 2, 7), 'open')
(701, 'loot', datetime.date(2021, 9, 5), 'open')
```

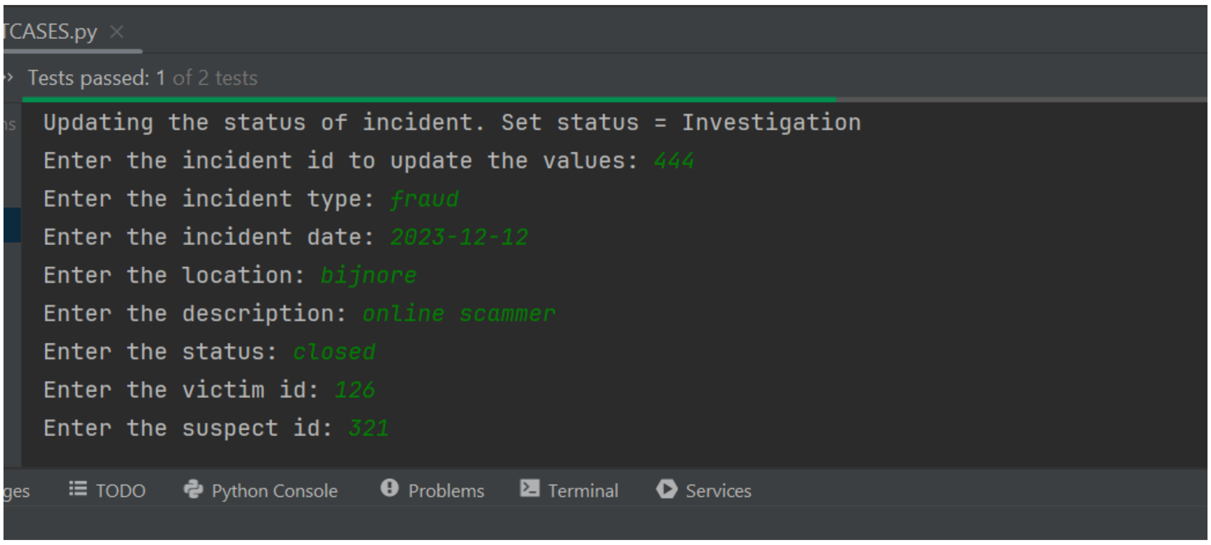
GAUTAM SHARMA

11) Testcases-for creating incident.

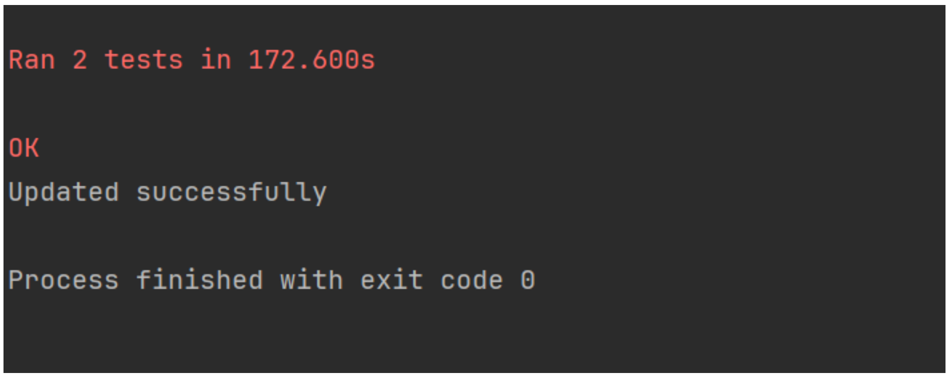


```
Create a new incident with incident id =
Enter the incident id: 444
Enter the incident type: atm theft
Enter the incident date: 2023-09-12
Enter the location: uttarpradesh
Enter the description: atm money was stolen
Enter the status: open
Enter the victim id: 888
Enter the suspect id: 999
```

13) Testcases-for updating an incident.



```
Updating the status of incident. Set status = Investigation
Enter the incident id to update the values: 444
Enter the incident type: fraud
Enter the incident date: 2023-12-12
Enter the location: bijnore
Enter the description: online scammer
Enter the status: closed
Enter the victim id: 126
Enter the suspect id: 321
```



```
Ran 2 tests in 172.600s

OK
Updated successfully

Process finished with exit code 0
```