PES UNIVERSITY

100 feet Ring Road, BSK 3rd Stage
Bengaluru 560085

Department of Computer Science and Engineering
B. Tech. CSE - 6th Semester
Jan – May 2023

UE20CS352
Object Oriented Analysis and Design with Java

Project Report

Forensic Evidence Management System

TEAM #:
PES1UG20CS633: Bhoomika
PES1UG20CS627: ARPITA KADAGAD
PES1UG20CS615: Nayana D Raj

Class of Prof.Bhargavi Mokashi

Problem Statement(synopsis)

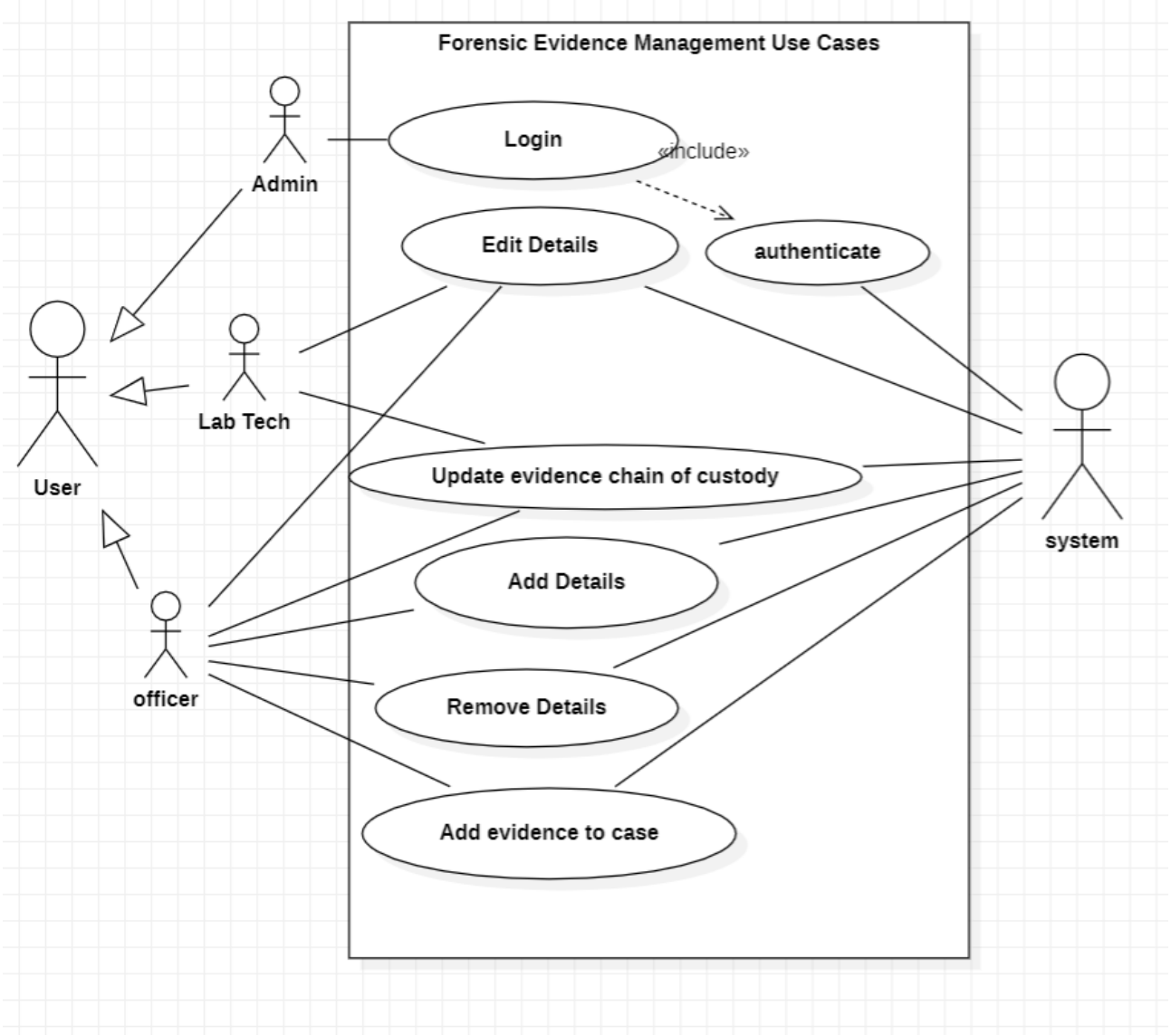
Forensic Evidence Management System is an Object-Oriented Analysis and Design project that aims to improve the management of forensic evidence in a law enforcement agency.The system will provide a centralized platform for storing, tracking and analyzing evidence data and will also enable the sharing of evidence information between different agencies.

The project will be divided into several phases, starting with requirements gathering and analysis.Once the requirements have been identified, the team will move on to the design phase, where they will create a use case diagram, class-diagram , state diagram and activity diagram.

After the design phase, the team will begin implementation, which will involve the development of the software using a suitable programming language i.e.,Java and framework i.e.,Spring and using spring tool suite IDE.The system will be designed to be scalable and secure, with appropriate access controls and data encryption mechanisms in place.

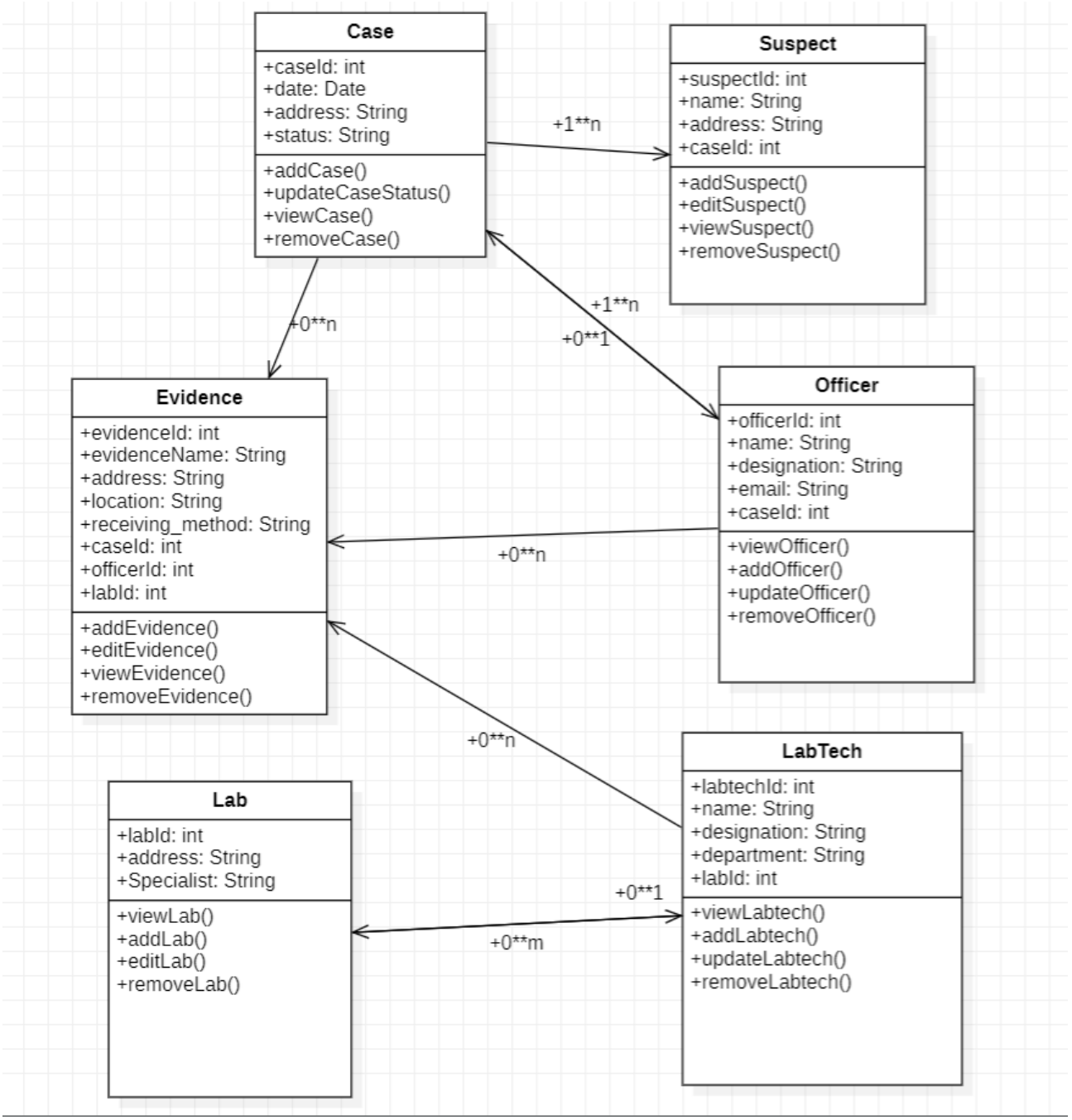
Overall, the Forensic Evidence Management System will provide a valuable tool for law enforcement agencies to manage and analyze forensic evidence data, leading to improved efficiency and effectiveness in solving crimes.

Use Case Model



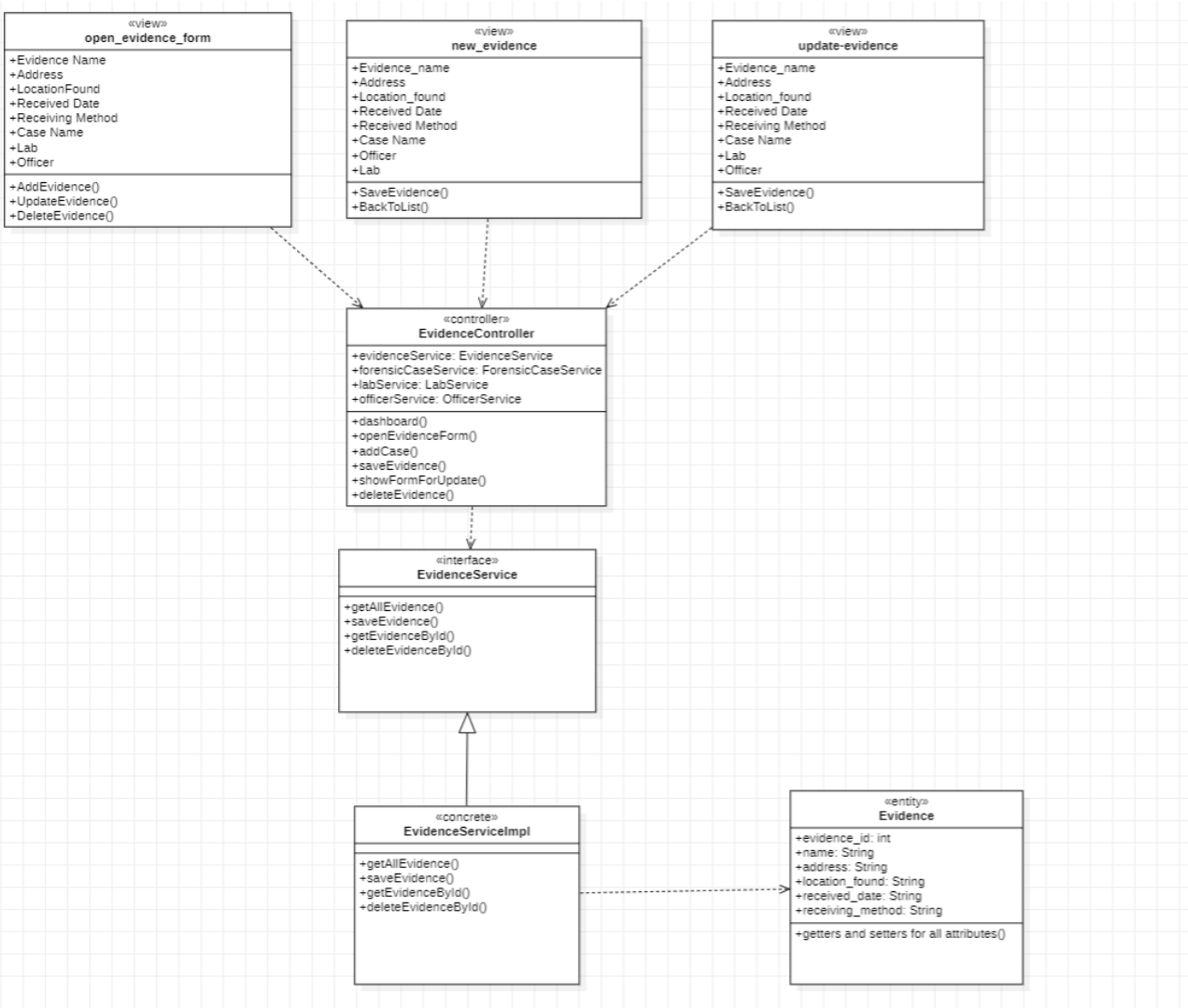
Expanded Use case (Edit Details for Evidence entity)

Class Model


Case
+caseId: int
+date: Date
+address: String
+status: String
+addCase()
+updateCaseStatus()
+viewCase()
+removeCase()
Suspect
+suspectId: int
+name: String
+address: String
+caseId: int
+addSuspect()
+editSuspect()
+viewSuspect()
+removeSuspect()
+1**n
+0**n
+1**n
+0**1
Evidence
+evidenceId: int
+evidenceName: String
+address: String
+location: String
+receiving_method: String
+caseId: int
+officerId: int
+labId: int
+addEvidence()
+editEvidence()
+viewEvidence()
+removeEvidence()
Officer
+officerId: int
+name: String
+designation: String
+email: String
+caseId: int
+viewOfficer()
+addOfficer()
+updateOfficer()
+removeOfficer()
+0**n
+0**n
LabTech
+labtechId: int
+name: String
+designation: String
+department: String
+labId: int
+viewLabtech()
+addLabtech()
+updateLabtech()
+removeLabtech()
Lab
+labId: int
+address: String
+Specialist: String
+viewLab()
+addLab()
+editLab()
+removeLab()
+0**1
+0**m

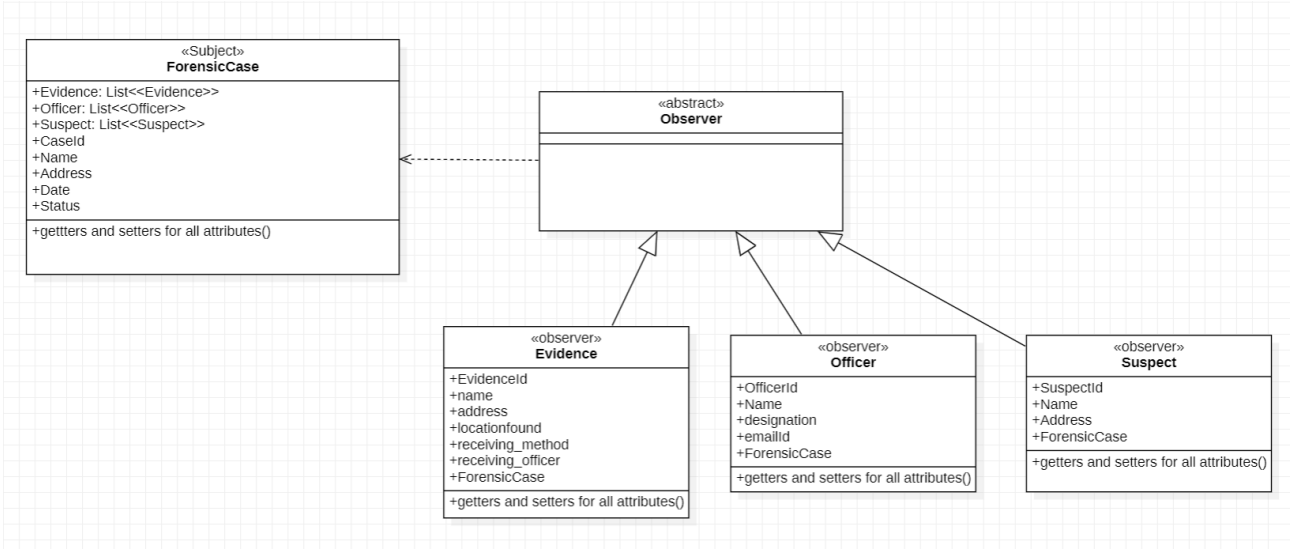
Updated Class Model

Dependency inversion principle



The same principle has been been used for all entities like Forensic case, Suspect, Officer,Lab, Labtech.

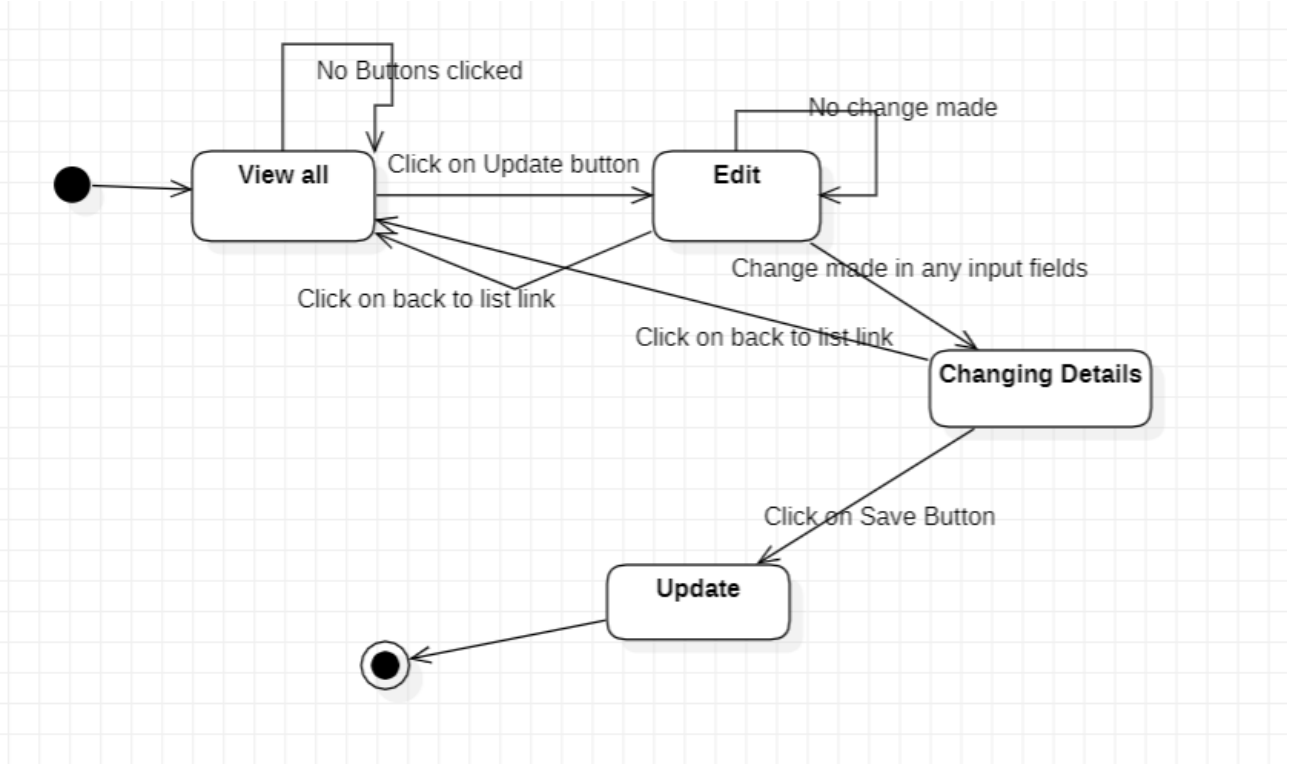
Observer Pattern



The same pattern has been used for relationship between Evidence, Labtech and Lab.

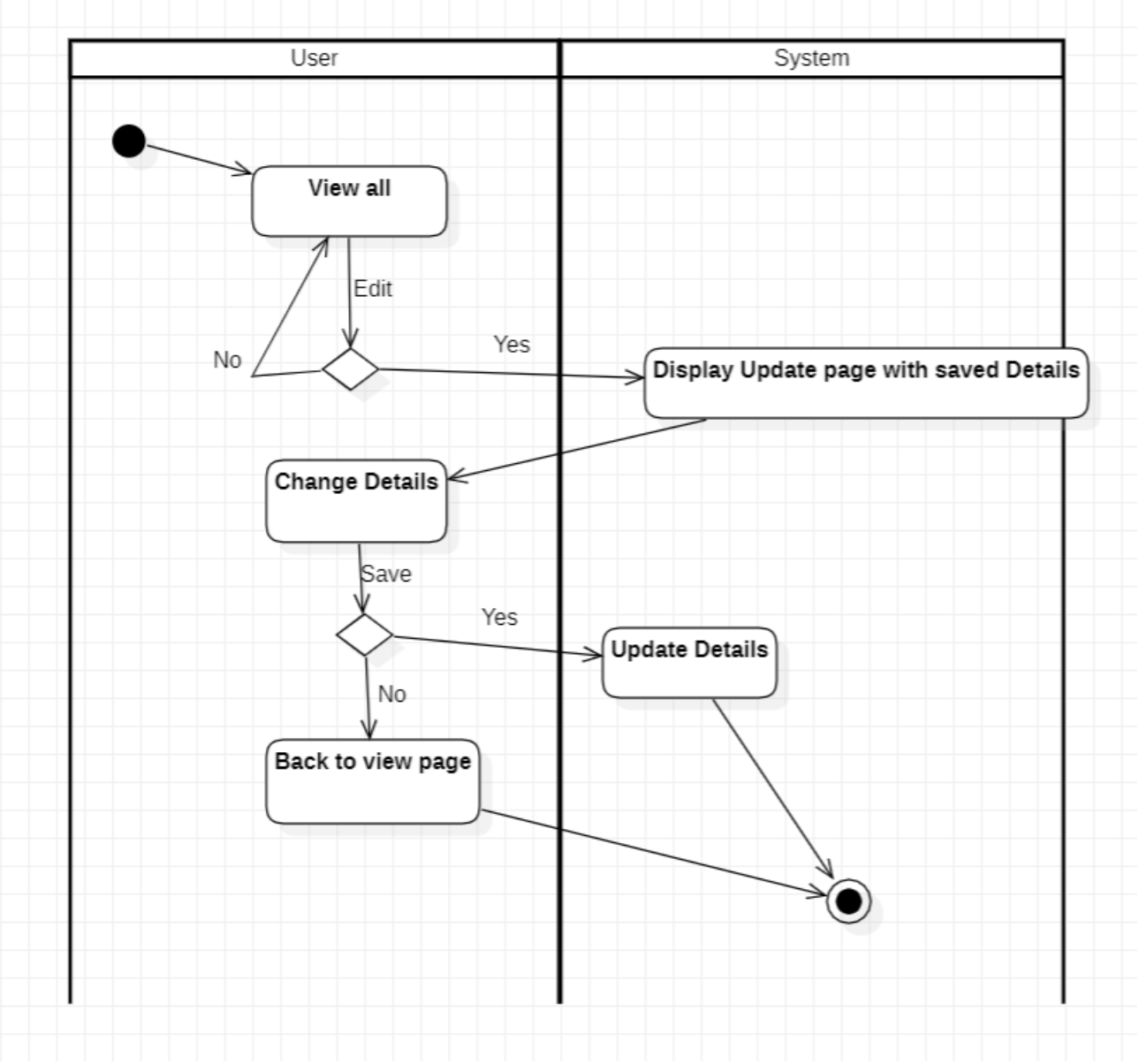
State Diagram:-

Edit Details use case



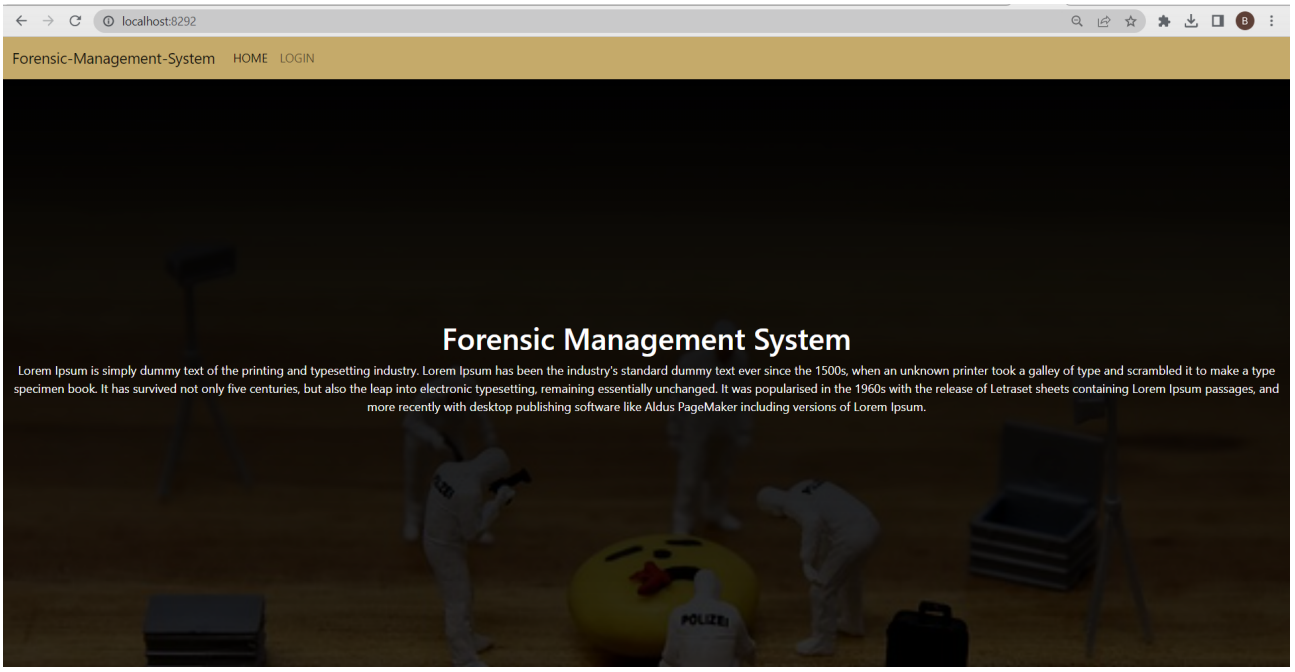
Activity Diagram:-

Edit Details use case

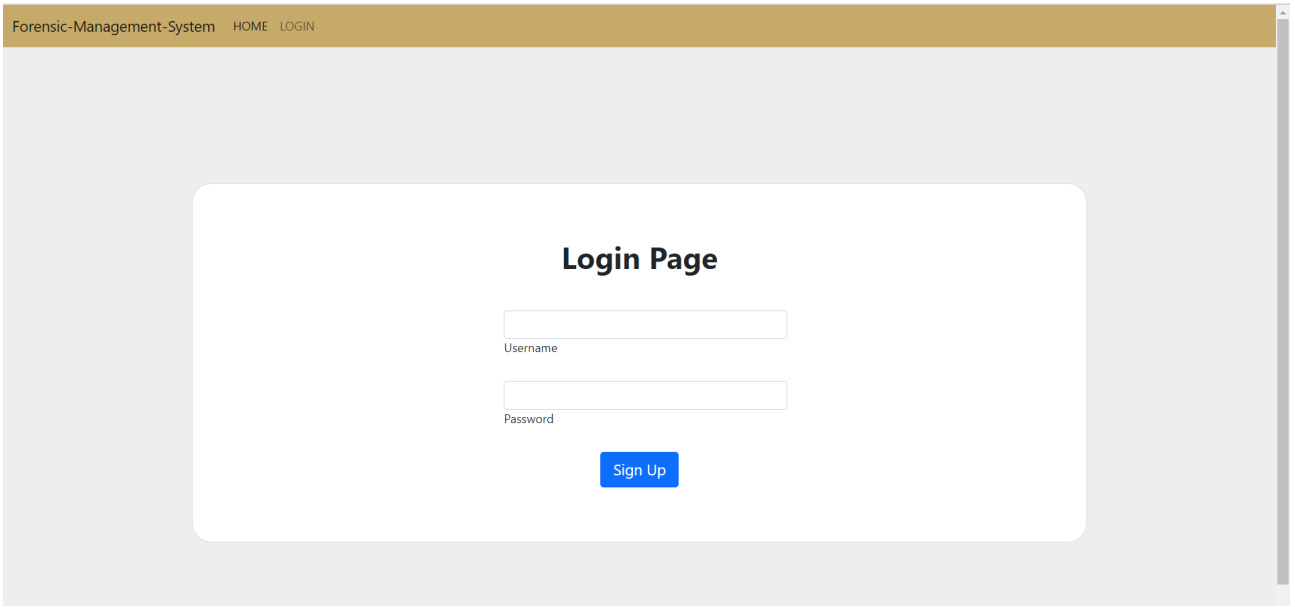


Output Screenshots:-

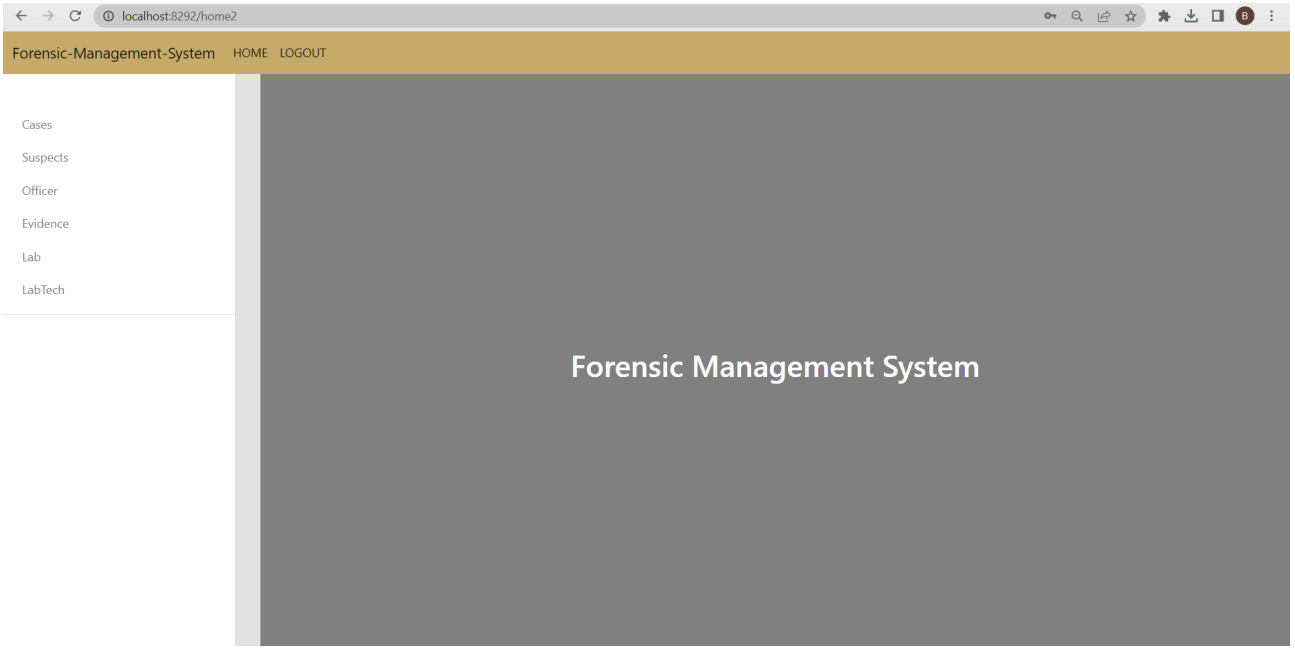
Home



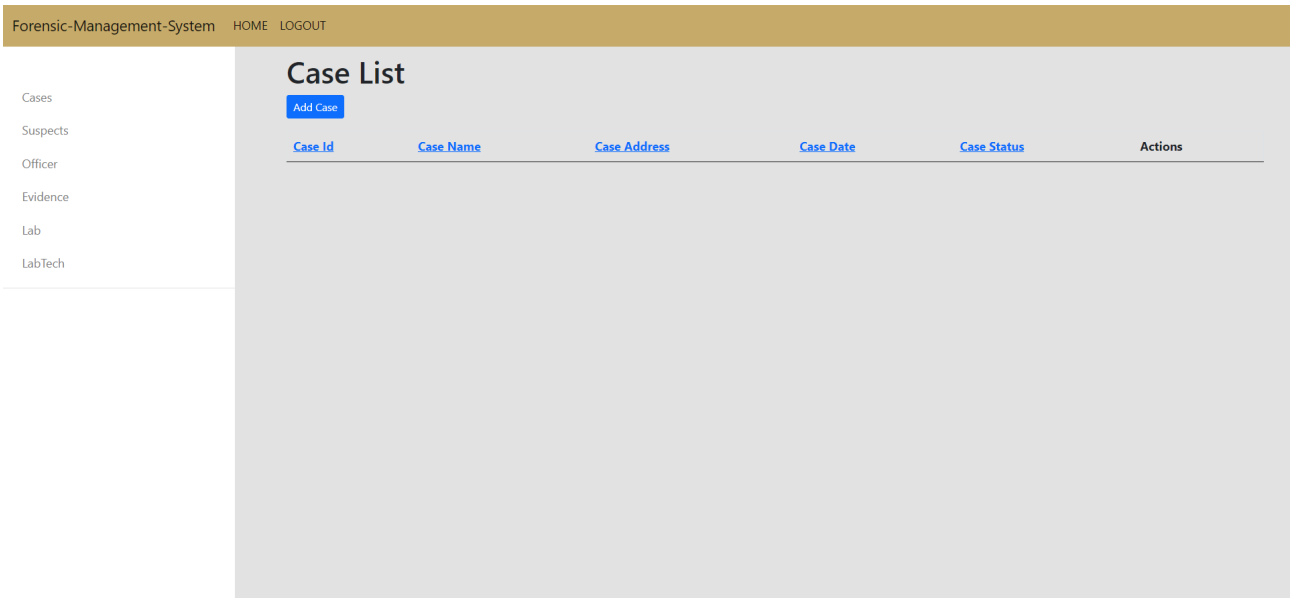
Login



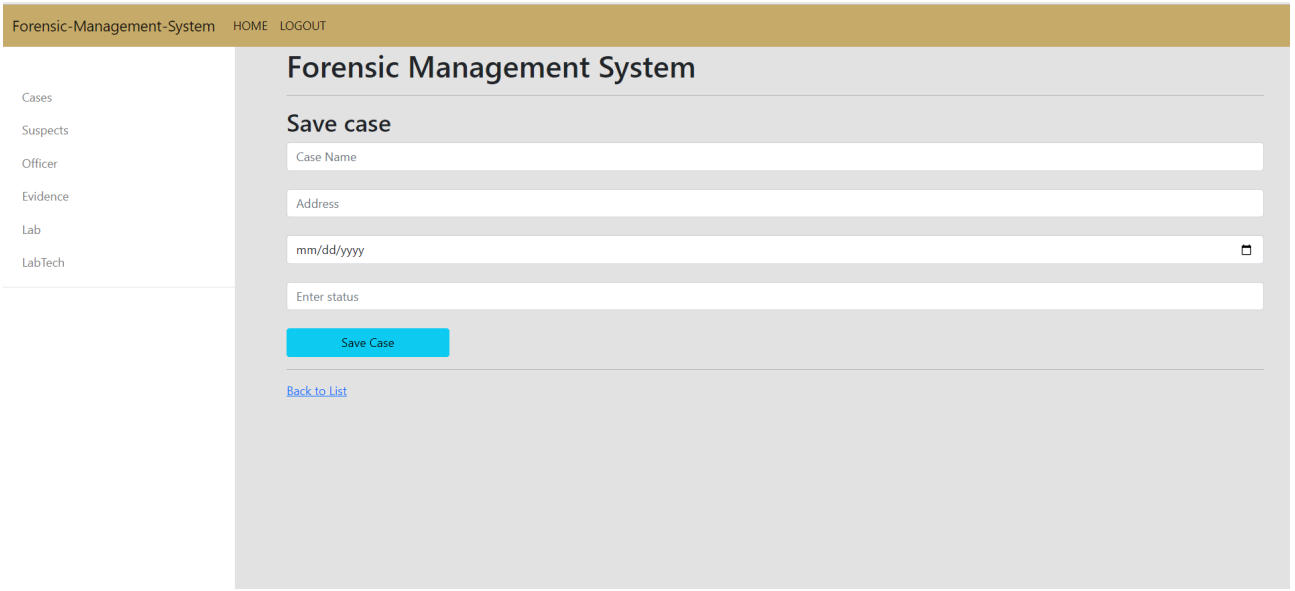
Dashboard



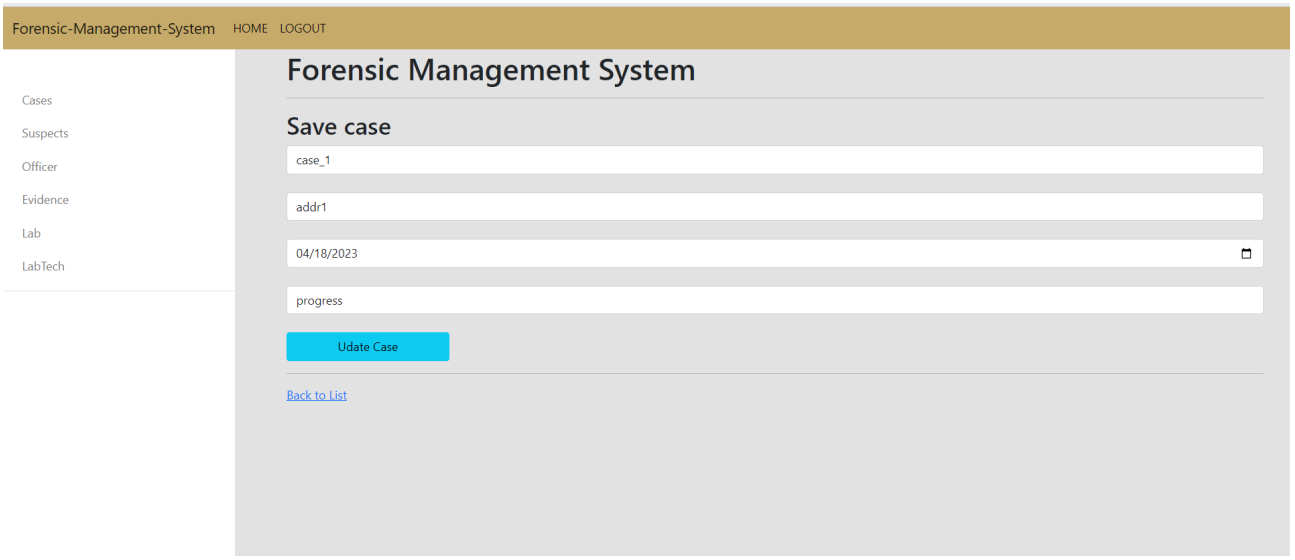
CRUD Operation:-



Add



Update



Delete

Database

```
mysql> use forensic_management
Database changed
mysql> show tables;
+--------------------------------+
| Tables_in_forensic_management |
+--------------------------------+
| evidence                       |
| forensic_case                  |
| lab                            |
| labtech                        |
| officer                        |
| suspect                        |
+--------------------------------+
6 rows in set (0.01 sec)
```

Tables_Updated

Forensic_case

```
mysql> select * from forensic_case;
+---------+---------+----------------------------+--------+----------+
| case_id | address | date                       | name   | status   |
+---------+---------+----------------------------+--------+----------+
|       2 | addr1   | 2023-04-17 18:30:00.000000 | case_1 | progress |
+---------+---------+----------------------------+--------+----------+
1 row in set (0.01 sec)
```

Suspect

```
mysql> select * from suspect;
+------------+----------------+-----------+---------+
| suspect_id | address        | name      | case_id |
+------------+----------------+-----------+---------+
|          1 | suspect_addr_1 | suspect_1 |       2 |
+------------+----------------+-----------+---------+
1 row in set (0.00 sec)
```

Lab

```
mysql> select * from lab;
+--------+------------+--------------------+
| lab_id | address    | department         |
+--------+------------+--------------------+
|      2 | lab_addr_1 | evidence_analysis  |
+--------+------------+--------------------+
1 row in set (0.01 sec)
```

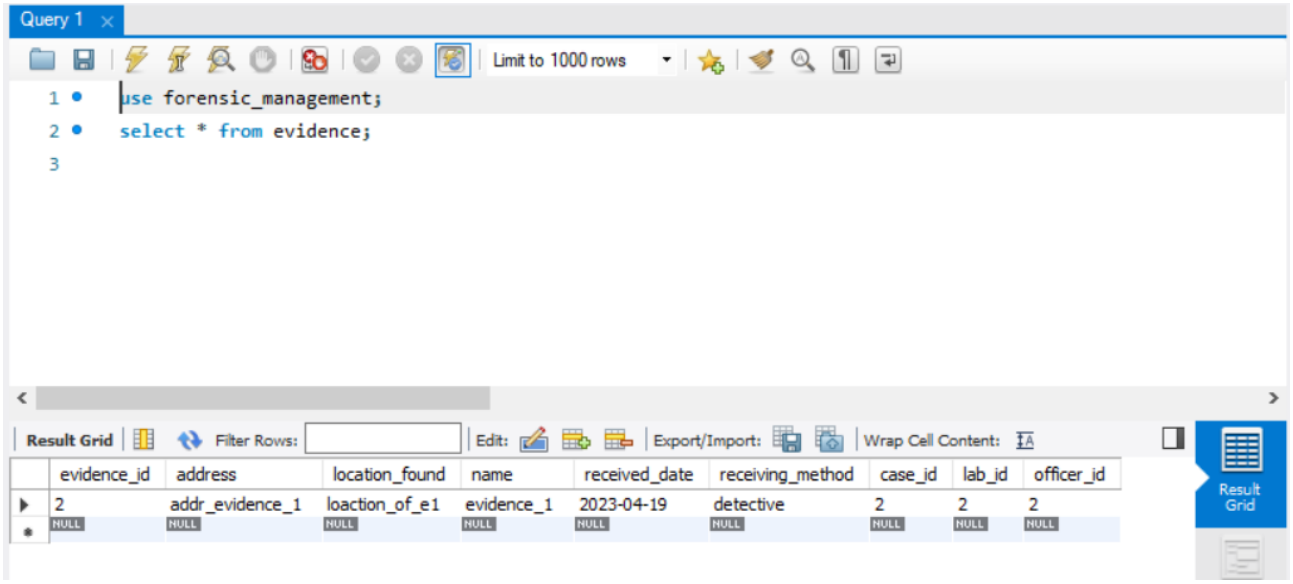
Labtech

```
mysql> select * from labtech;
+------------+------------------+------------+----------+--------+
| labtech_id | department       | designation | name     | lab_id |
+------------+------------------+------------+----------+--------+
|          1 | evidence_analysis | junior_tech | labtech_1 |      2 |
+------------+------------------+------------+----------+--------+
1 row in set (0.00 sec)
```

Officer

```
mysql> select * from officer;
+------------+-------------+---------------+----------+---------+
| officer_id | designation | email         | name     | case_id |
+------------+-------------+---------------+----------+---------+
|          2 | Senior      | abc@gmail.com | officer_1 |       2 |
+------------+-------------+---------------+----------+---------+
1 row in set (0.00 sec)
```

Evidence

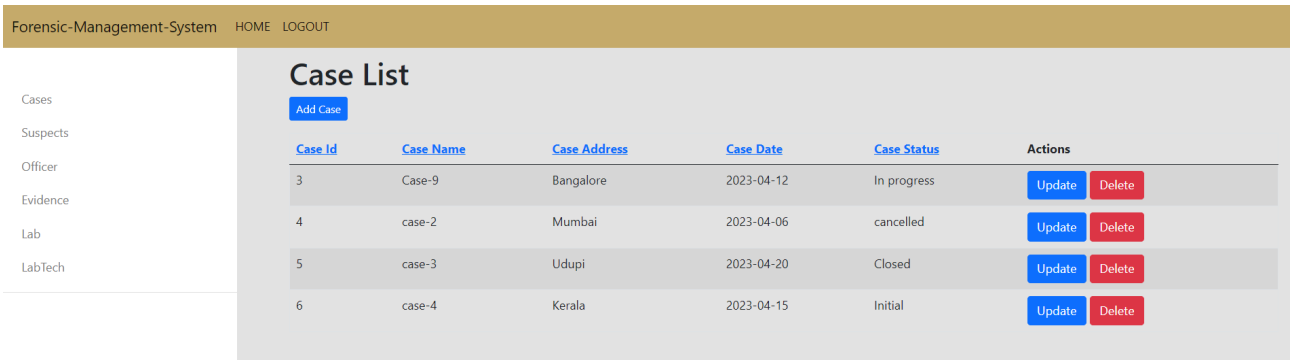


Design Pattern

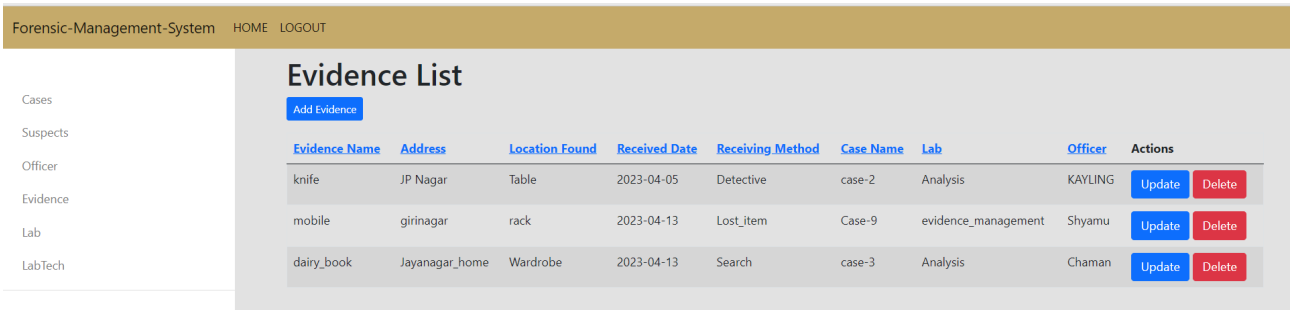
Observer Pattern

For case-id initially case_name is case-9 after updating case_name to case-10 in case table it will be updated in evidence table also

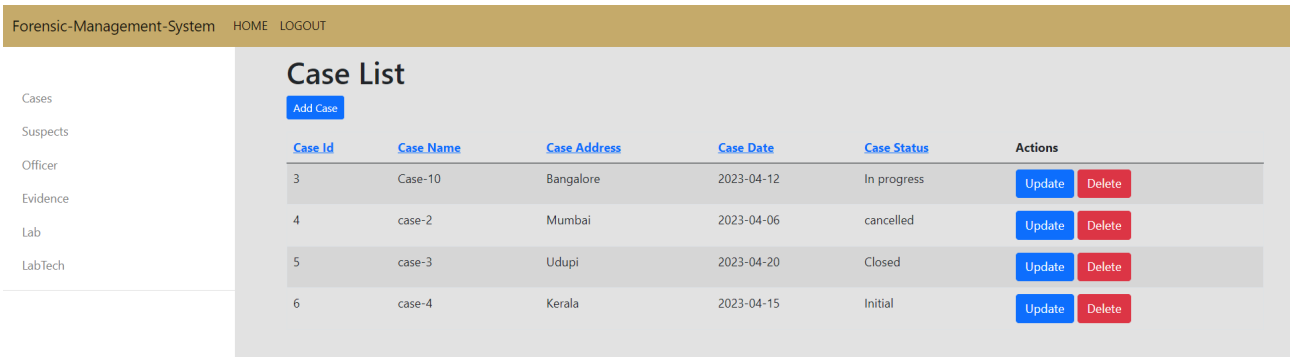
Before updation case table:-



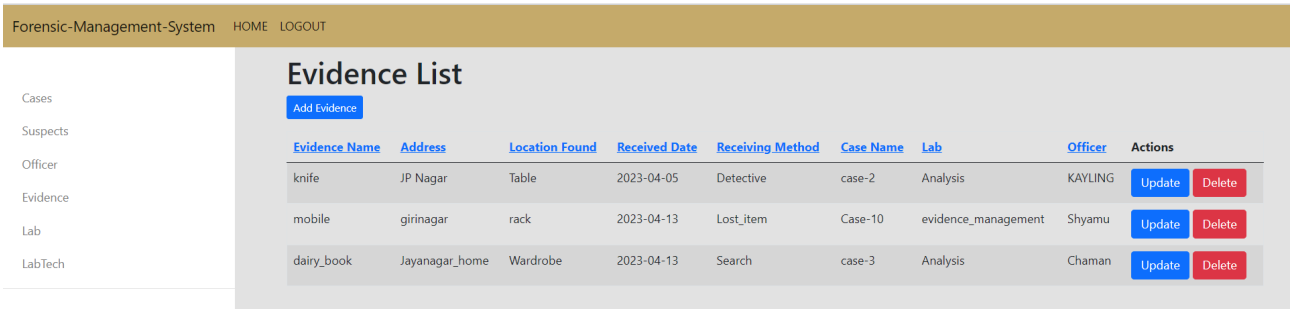
Before updation evidence table:-



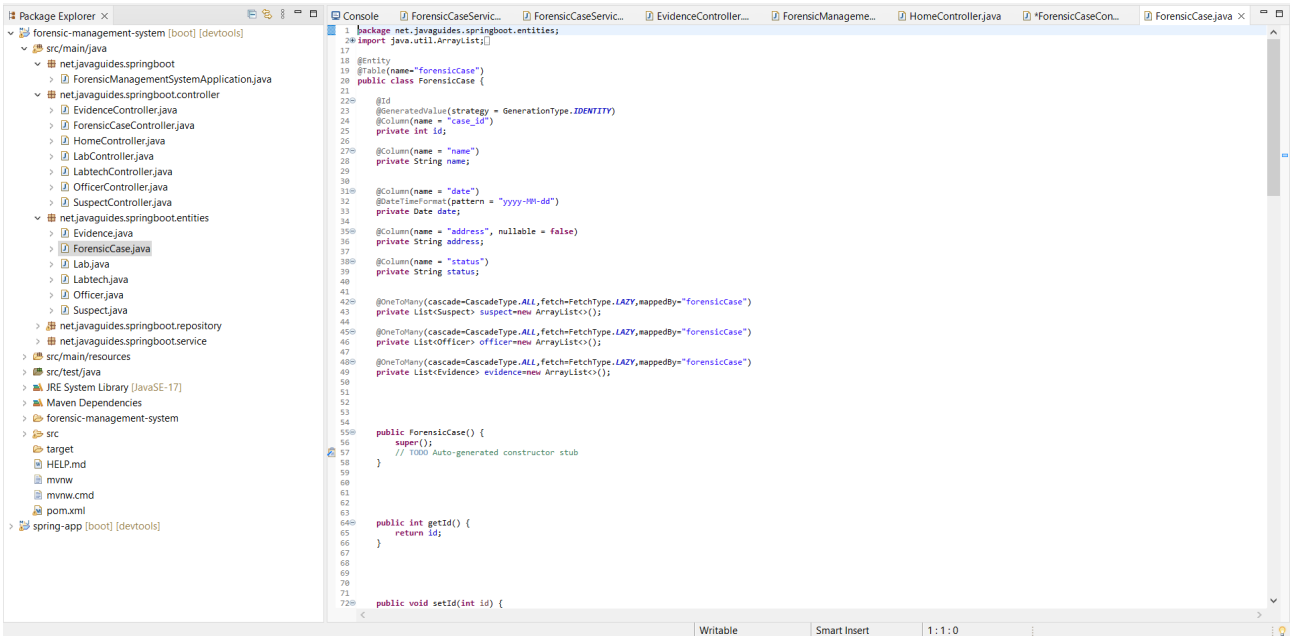
After updation in case table:-

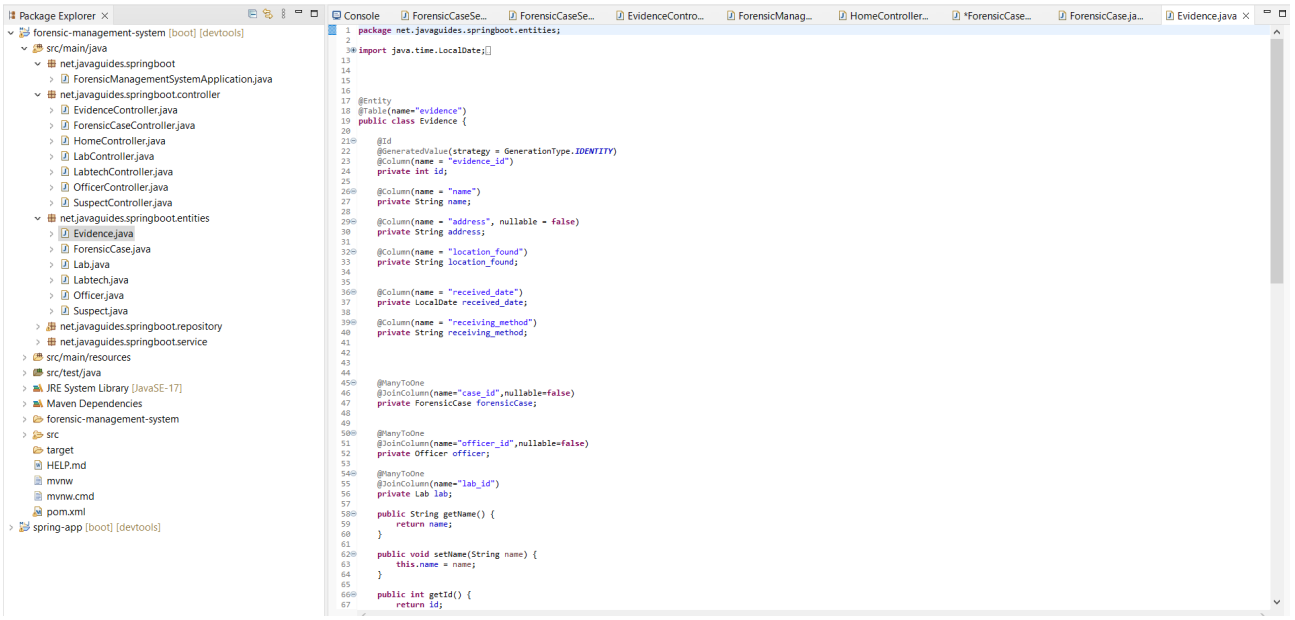


After updation in evidence table:-



Code for observer pattern:-



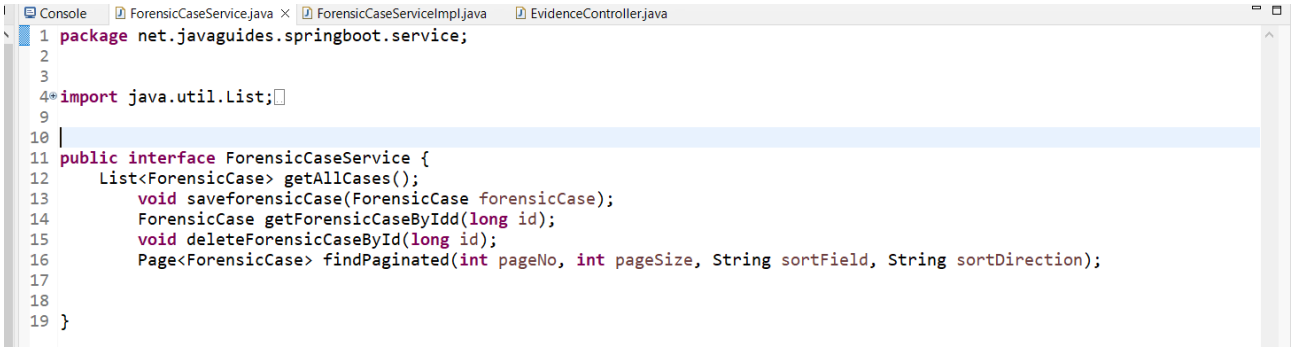


Dependency Inversion principle:-

The Dependency Inversion Principle (DIP) states that high level modules should not depend on low level modules; both should depend on abstractions.

ForensicCaseController and ForensicCaseServiceImpl are dependent on ForensicCaseService Interface.

ForensicCaseService.java



```java
package net.javaguides.springboot.service;


import java.util.List;

public interface ForensicCaseService {
    List<ForensicCase> getAllCases();
        void saveforensicCase(ForensicCase forensicCase);
        ForensicCase getForensicCaseByIdd(long id);
        void deleteForensicCaseById(long id);
        Page<ForensicCase> findPaginated(int pageNo, int pageSize, String sortField, String sortDirection);


}
```

ForensicCaseServiceImpl.java

```java
package net.javaguides.springboot.service;

import java.util.List;

@Service
public class ForensicCaseServiceImpl implements ForensicCaseService {

    @Autowired
    private CaseRepository caseRepository;

    @Override
    public List<ForensicCase> getAllCases() {
        return caseRepository.findAll();
    }

    @Override
    public void saveforensicCase(ForensicCase forensicCase) {
        this.caseRepository.save(forensicCase);
    }

    @Override
    public ForensicCase getForensicCaseByIdd(long id) {
        Optional<ForensicCase> optional = caseRepository.findById(id);
        ForensicCase forensicCase = null;
        if (optional.isPresent()) {
            forensicCase = optional.get();
        } else {
            throw new RuntimeException(" ForensicCase not found for id :: " + id);
        }
        return forensicCase;
    }

    @Override
    public void deleteForensicCaseById(long id) {
        this.caseRepository.deleteById(id);
    }

    @Override
    public Page<ForensicCase> findPaginated(int pageNo, int pageSize, String sortField, String sortDirection) {
        Sort sort = sortDirection.equalsIgnoreCase(Sort.Direction.ASC.name()) ? Sort.by(sortField).ascending() :
            Sort.by(sortField).descending();

        Pageable pageable = PageRequest.of(pageNo - 1, pageSize, sort);
        return this.caseRepository.findAll(pageable);
    }
}
```

ForensicCaseController.java