Software Requirements and Design Document

for

Society360

Prepared by Hammad Muhammad Siddique & Muhammad Haziq Naeem & Muhammad Bilal

Passion Associates

Wednesday 27th November 2024

Table of Contents

1. Introduction

1.1 Purpose

The product is designed to digitalize the management of a city. The product allows residents as well as administrators to manage the city digitally. Residents are able to request services such as recreation, healthcare or any other public service whereas the admins are able to handle those requests and schedule any infrastructure maintenances if needed.

1.2 Product Scope

The product is designed at the city level. It focuses on how a city can be managed digitally. It aims to help society become digital and provide a one-window system where all residents can access any required service

1.3 Title

Society360

1.4 Objectives

- *Provide a single platform which would help residents to access all the services in a city pr society*
- *Allow residents to report any problems they face.*
- *Make the management of a city or society easy by providing a digital platform.*

1.5 Problem Statement

Managing cities and societies is a huge that especially for large countries. There are a ton of different services that can be required by the residents, and they can face a variety of different problems. For large populations, it becomes even more complicated. Therefore, Society360 comes into play providing a single platform for both residents and admins to be able to live in and manage the city or society efficiently.

2. Overall Description

2.1 Product Perspective

This product is a stand-alone product which has no supersystems or subsystems. It may be worked on and improved in future what no supersystem or subsystem will be added. So this product is stand-alone and fulfills a defined set of purposes.

2.2 Product Functions

- *The product is designed to provide a platform for making city or society management easy.*
- *There are two types of users: admins or residents*
- *Residents are also to request any services or report any problems*
- *Admins can handle those requests and problems, where they can either approve, reject or ignore them.*
- *The residents can vote digitally on the city or society initiatives they prefer should be worked on*

2.3 List of Use Cases

- *Register Property*
- *Request Public Service*
- *Digital Voting*
- *Reserve Recreation Facility*
- *Report Crime or Incident*
- *Submit Public Feedback*
- *Pay Utility Bill*
- *Acquire Business License*
- *Manage Institutes*
- *Manage Healthcare*
- *Schedule Infrastructure Maintenance*

2.4 Extended Use Cases

Use Case Name: Reserve a Recreation Facility

- Scope: Recreation Management System
- Level: User goal
- Primary Actor: Resident
- Stakeholders and Interests:

o Resident: Wants to reserve the facility for personal or community use.
o Recreation Administrator: Wants to ensure facility is booked fairly and is maintained properly.

- Preconditions:

o The recreation facility must be available for booking.
o Resident must be logged in.

- Postconditions:

o Success: Reservation is successfully made and confirmed.
o Failure: Reservation is not made due to unavailable facility or system error.

- Main Success Scenario:

1. Resident logs into the system.
2. Resident selects the facility and desired date.
3. System checks availability.
4. System confirms availability.
5. Resident submits the reservation request.
6. System notifies the Recreation Administrator.

7. Recreation Admin approves the request.
8. Resident receives confirmation of the reservation.

- Extensions:
- 3a. Facility is already booked.
- - 3a1. System informs resident and suggests alternative dates.
- 3b. Resident is not eligible to reserve.
- - 3b1. System denies booking and displays reason

Use Case Name:Register Property

- Scope:Property Management System
- Level:User goal
- Primary Actor:Resident
- Stakeholders and Interests:

o Residence Admin: Wants accurate records of properties.
o Resident: Wants to register property for legal and regulatory reasons.

- Preconditions:

o Resident must provide all necessary property details.

- Postconditions:

o Success: Property is successfully registered and confirmed.
o Failure: Property is not registered due to missing or incorrect details.

- Main Success Scenario:

1. Residence Admin logs into the system.
2. Residence Admin submits the property details.
3. System verifies the details.
4. System registers the property.
5. Resident is notified of successful registration.
6. System updates the property database.

- Extensions:
- 3a. Property details are incomplete.
- - 3a1. System requests missing details.
- 3b. Property already registered.
- - 3b1. System notifies the Resident of existing registration

Use Case Name:Schedule Infrastructure Maintenance

- Scope:City Maintenance Management System
- Level:User goal
- Primary Actor:City Administrator
- Stakeholders and Interests:

o City Administrator: Wants to schedule maintenance for infrastructure upkeep.
Resident: Expects minimum disruption during maintenance.

- Preconditions:

o Maintenance needs have been identified.

- Postconditions:
- Success: Maintenance is scheduled and communicated to residents.
- Failure: Maintenance is not scheduled due to conflicts or errors.
- Main Success Scenario:

1. City Admin logs into the system.
2. City Admin selects the infrastructure to be maintained.
3. City Admin selects a time slot.
4. System checks for conflicts with other services.
5. System schedules the maintenance.
6. System notifies affected residents.

- Extensions:
- 2a. Conflicting maintenance schedule.
- - 2a1. System suggests alternative times.
- 4a. Maintenance is delayed.
- - 4a1. System notifies residents of new schedule.

Use Case Name:Submit Public Feedback

- Scope:Public Feedback System
- Level:User goal
- Primary Actor:Resident
- Stakeholders and Interests:

o Resident: Wants their feedback to be heard by authorities.
o City Administrator: Wants to collect and analyze public feedback to improve services.

- Preconditions:

o The resident must be logged into the system.

- Postconditions:

o Success: Feedback is submitted and reviewed by City Admin.
o Failure: Feedback submission fails due to system error or incomplete form

- Main Success Scenario:

1. Resident logs into the system.
2. Resident selects the "Submit Feedback" option.
3. Resident completes the feedback form.
4. System verifies the information.
5. System submits the feedback.
6. City Admin receives and reviews the feedback.
7. System confirms feedback submission to the Resident.

- Extensions:
- 3a. Incomplete feedback form.
- - 3a1. System prompts Resident to complete missing fields

Use Case Name: Pay Utility Bills

Scope : Utility Billing System

Level : User goal
Primary Actor: Resident
Stakeholders and Interests:
- Resident: Wants to pay utility bills promptly.
- Utility Admin: Wants to receive and process payments..

Preconditions:

- Bill must be available in the system. Resident must have sufficient funds.

Postconditions:

- Utility bill is paid, and payment is recorded..

Main Success Scenario:
1. Resident logs into the system.
2. Resident views outstanding utility bills.
3. Resident selects a bill and makes payment.
4. System processes payment.
5. Resident receives payment confirmation.
6. Utility Admin records payment in the system.

Extensions:
1. Insufficient funds.
2. System notifies Resident of payment failure.

Use Case Name: Acquire Business License
Scope: System Level Scope
Level: User goal
Primary Actor : Business Owner
Stakeholders and Interests:
- Business Owner: Wants smooth and successful approval of the business license
- Business Administrator: Wants fast and efficient processing in providing business licenses

Preconditions: The business administrator is authorized and verified. And the business owner is verified
Postconditions: The business license has been successfully approved and acquired
Main Success Scenario:
1. The business owner opens license application form and enters all the required details
2. System processes the application and puts it on waiting-approval list
3. Business Administrator logs into the system and opens business license applications
4. Business Administrator approves the application and provides license to the business owner

Extensions:

- The business admin rejects the business application

- The business owner is unable to submit the application due to system error

Use Case Name: Request New Public Service
Scope: System Level Score
Level: User goal
Primary Actor : Resident
Stakeholders and Interests:
- Resident: Needs fast provision of a new service
- Services Administrator: Needs to ensure that every service request is entertained on time
- Service Provider: Needs to provide efficient service to the resident
Preconditions: The Services admin is authorized, and the service provider is approved
Postconditions: The requested service has been successfully provided to the resident
Main Success Scenario:
1. The resident accesses the services portal
makes a service request
2. The system processes the request and adds
it to the service requests list
3. The services admin logs into the system
and checks for service requests
4. The services admin contacts the service
provider and assigns him the service request
5. The service provider provides the
requested service to the resident
Extensions

- The service provider is busy or not available, so service request is rejected to delayed
- The resident is unable to request the service due to system error

Use Case Name: Digital Voting for City Initiatives
Scope: System
Level: User goal
Primary Actor: Resident
Stakeholders and Interests:
- Resident: Votes on which City initiatives should be taken
- Electoral Head: Ensures that the voting is conducted smoothly
- City Manager: Wants public feedback on which initiatives should be taken for the city
Preconditions: All the voters are registered city residents. And the Electoral Head is unbiased
Postconditions: The voting has been conducted smoothly and a fair result on which initiatives should be taken has been obtained
Main Success Scenario:
1. The city manager drafts a list of different
city initiatives

2. The city manager organizes the voting and
appoints an electoral head
3. The electoral head obtains a list of all
registered voters
4. The electoral head setups up and opens
portal for digital voting
6. After the voting ends, the system counts
votes and displays result to the electoral head
5. The residents vote for their desired
initiatives
7. The electoral head announces the results
of the elections to the public and forwards
them to the city manager
Extensions:

- The voting is rigged by the electoral head
- The voting system breaks and people are unable to vote

Use Case Name: Report Traffic Incident
Scope: System
Level: User goal
Primary Actor: Traffic Administrator
Stakeholders and Interests:
- Traffic Administrator: Wants that the incident is handled correctly
- Resident: Wants accurate and timely reporting of the traffic incident
- Safety Worker: Provides help in resolving the traffic incident

Preconditions: The resident and the traffic admin are registered
Postconditions: The traffic incident has been handled correctly
Main Success Scenario:
1. The resident logs into the portal and
initiates traffic incident reporting
2. The resident provides the details of the
traffic incident
3. The system processes the details and sends
them to the traffic administrator
4. The traffic administrator views the traffic
incident reports.
5. The traffic administrator appoints safety
workers to resolve the traffic incident
6. The safety workers reach on-site and
handle the traffic incident
Extension:

- There are no available safety workers, so the traffic incident remains unsolved
- There is some misreporting in the traffic incident so it is not handled correctly

- The resident is unable to report the traffic incident due to system issues

Use Case Name: Manage Institutions
Scope: System
Level: User goal
Primary Actor: Education Administrator
Stakeholders and Interests:
- Education Admin: Ensure smooth running and maintenance of all educational institutions
- Students: Good teaching staff and on-time curriculum completion
- Teachers: Smooth administration of institutions

Preconditions: The education administrator is registered
Postconditions: Smooth management of all educational institutions
Main Success Scenario:
1. The education admin logs into the system
and checks for any issues reported
2. The education admin forwards any
complaints or issues reported to the
appropriate departments for resolution
3. The departments appoint staff members to
address different issues
4. The staff members investigate and work on
the reported issues

Extension:
- The institutions are unable to report the issues due to system errors
- There is misreporting of certain issues which leads to them not being handled correctly

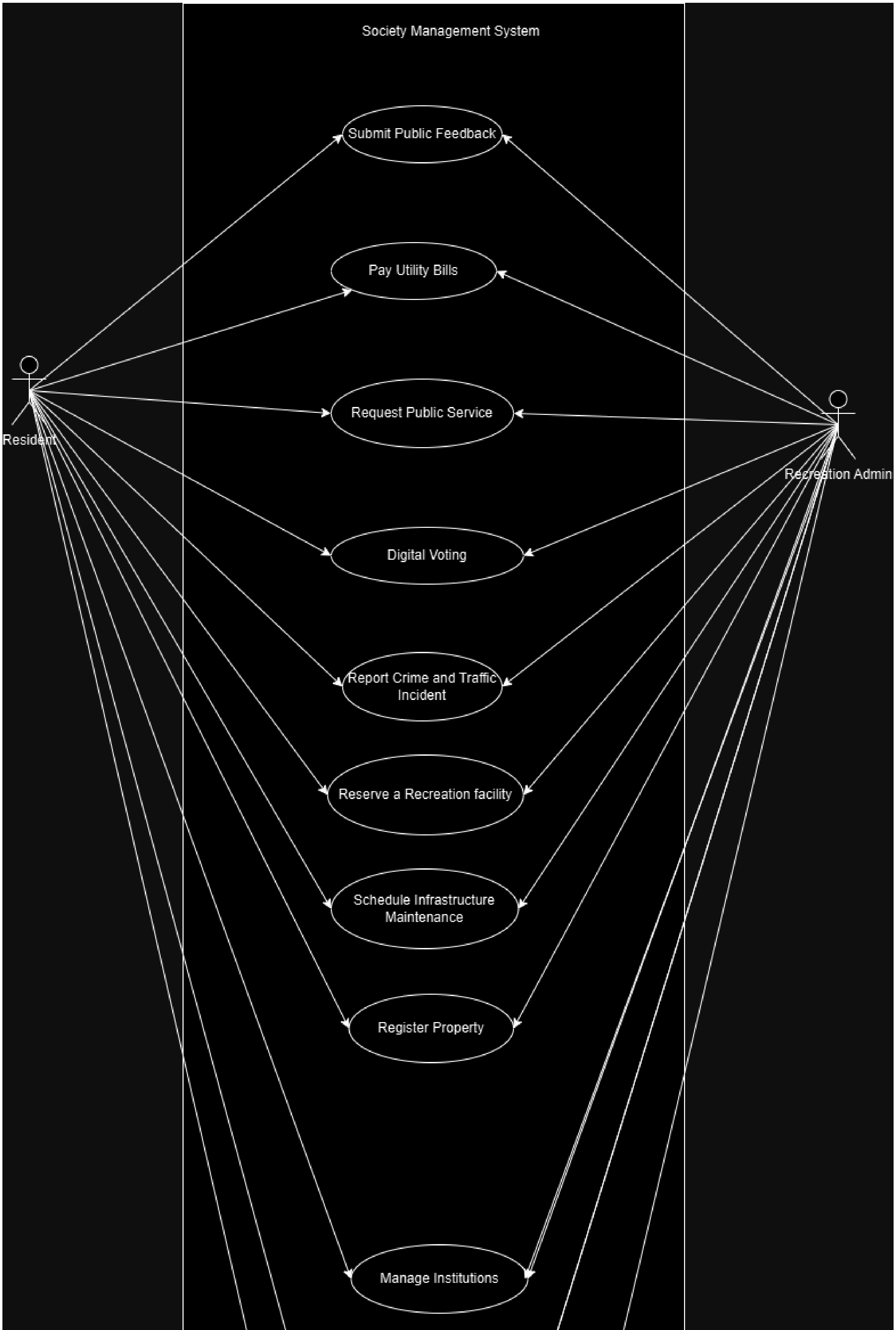
Use Case Name: Manage Healthcare
Scope: System
Level: User goal
Primary Actor: Healthcare Administrator
Stakeholders and Interests:
- Education Admin: Ensure smooth running and maintenance of all healthcare centers
- Doctors: Expect effective and smooth management of all healthcare centers
- Patients: Want fast and efficient healthcare services

Preconditions: The education admin is authorized, and the healthcare center is approved
Postconditions: All the healthcare centers are being managed effectively
Main Success Scenario:
1. The healthcare admin logs into the system
and checks for any issues reported
2. The healthcare admin forwards any
complaints or issues reported to the
appropriate departments for resolution

3. The departments appoint staff members to
address different issues
4. The staff members investigate and work on
the reported issues
Extension:

- The healthcare centers are unable to report the issues due to system errors
- There is misreporting of certain issues which leads to them not being handled correctly

2.5 Use Case Diagram


Society Management System
Submit Public Feedback
Pay Utility Bills
Request Public Service
Digital Voting
Report Crime and Traffic Incident
Reserve a Recreation facility
Schedule Infrastructure Maintenance
Register Property
Manage Institutions
Resident
Recreation Admin

3. Other Nonfunctional Requirements

3.1 Performance Requirements

There are no special performance requirements. The product is able execute on the most basic of systems since the product goes not require much computational power

3.2 Safety Requirements

There are no safety requirements

3.3 Security Requirements

The product is secure and built on a robust system which ensures that all the data is secure and there is no chances of data corruption or data loss.

3.4 Software Quality Attributes

The product is robust and well-tested. The product offers ease of use over easy of learning. Despite that the product is complicated to learn, it is even easier to use, because of its simple UI and well-structured user-experience. Because of a lack of instructions in the product on how to use it, there may arise some difficulties in its usage. Proper error handling has been implemented so the product is robust.

3.5 Business Rules

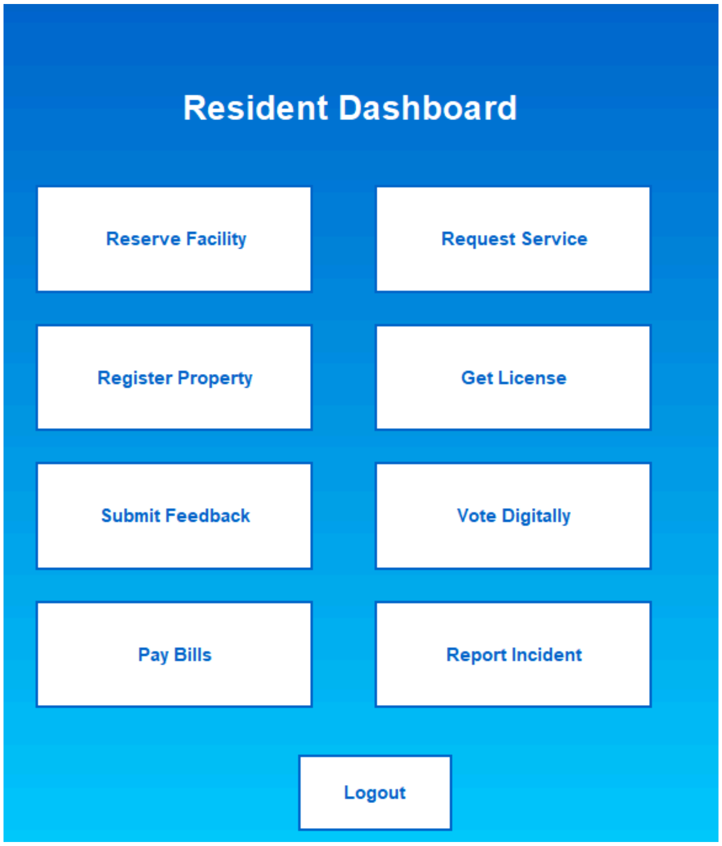
There are no specific business rules. The program is executable on any operating system that runs normally under normal circumstances.

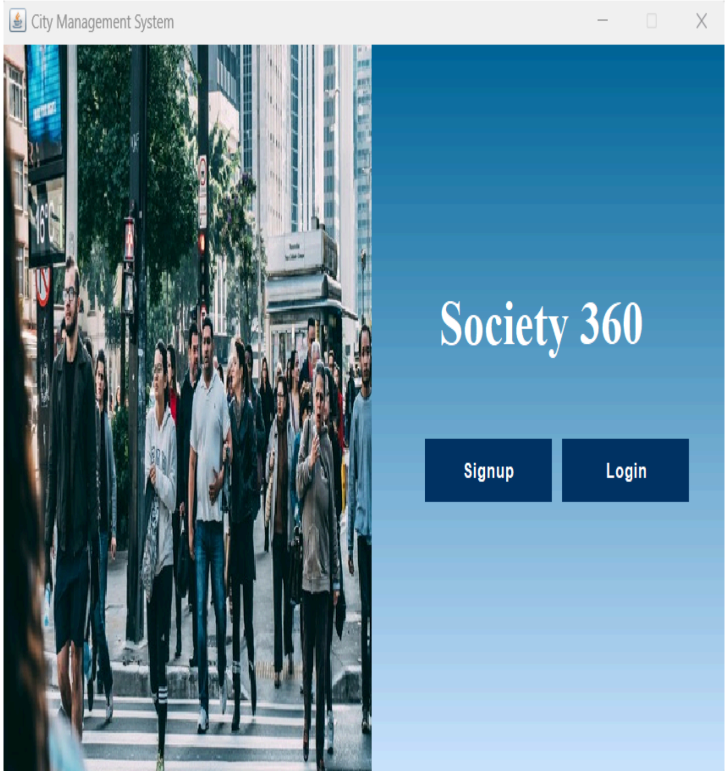
3.6 Operating Environment

The project is created in Java using the Swing library and MySQL database is used. Since Java and MySQL are supported by all operating systems, the product can operate successfully in all normal environments.

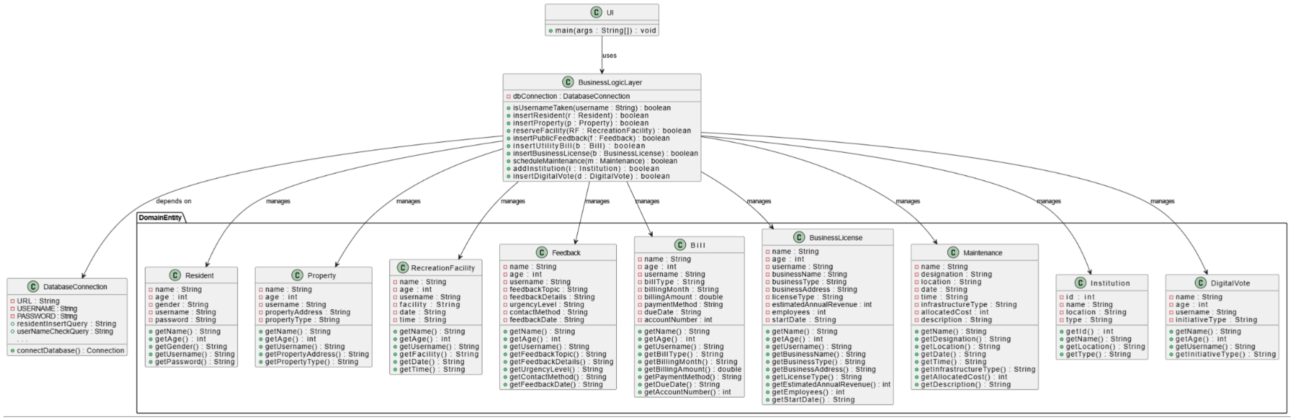
3.7 User Interfaces

The product uses friendly and simple user interfaces. The UI components used include buttons, forms and lists. Below are some of the UI screens of the product:

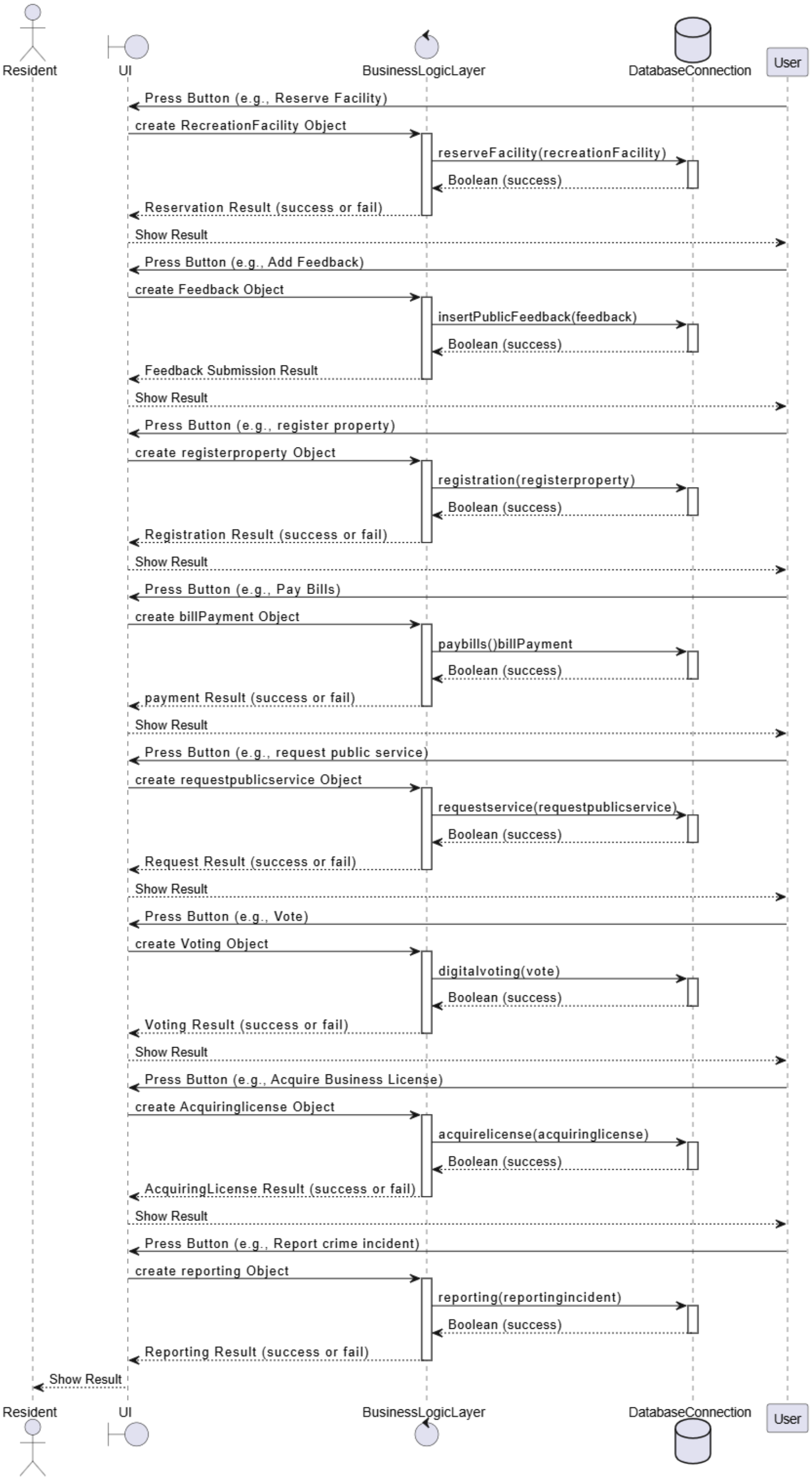


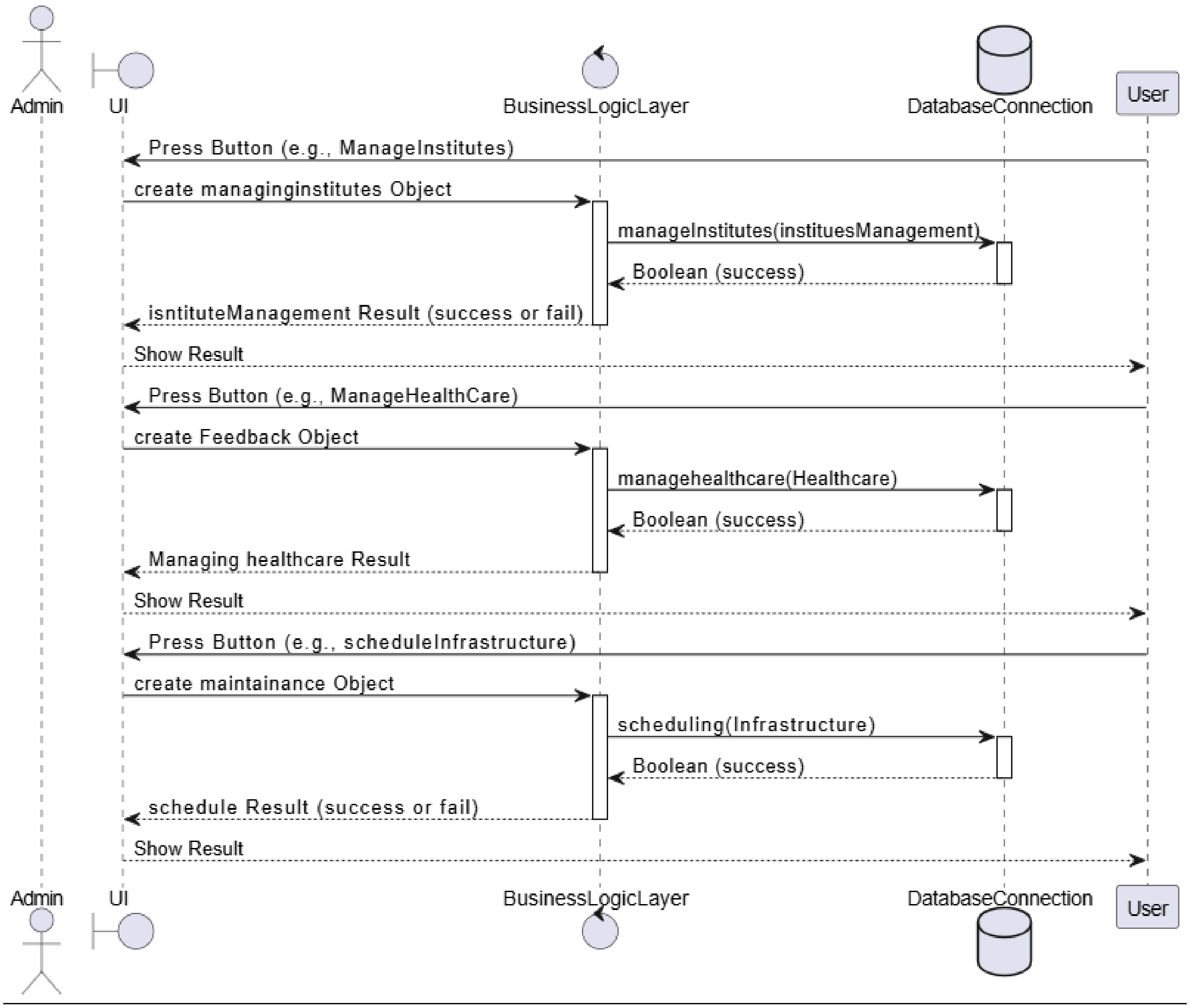


4. Domain Model

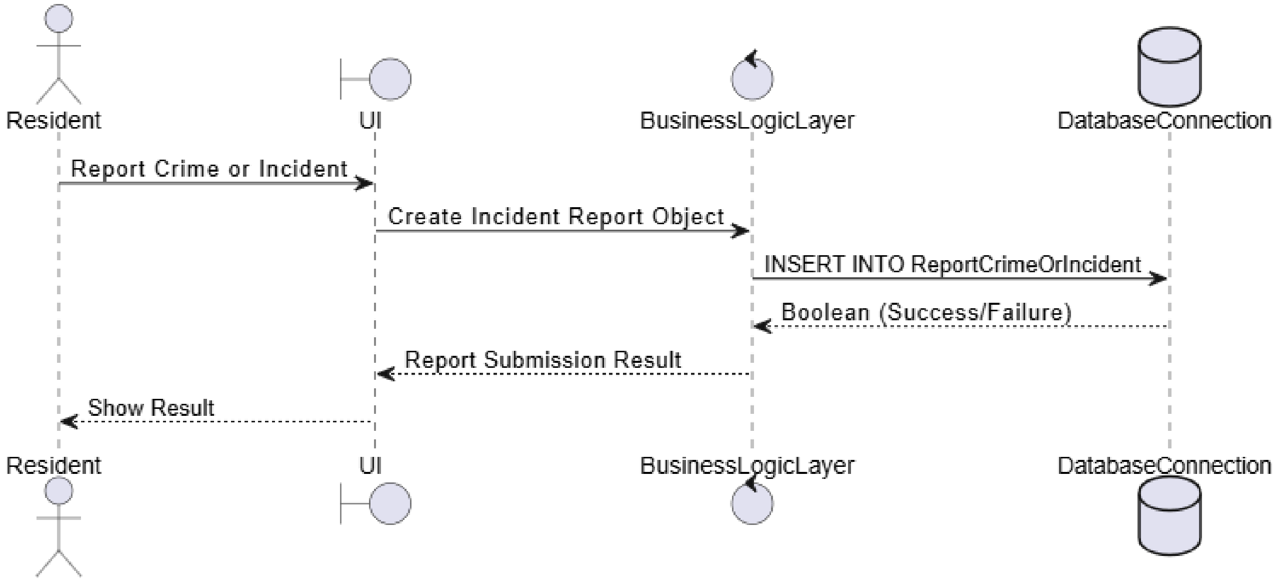


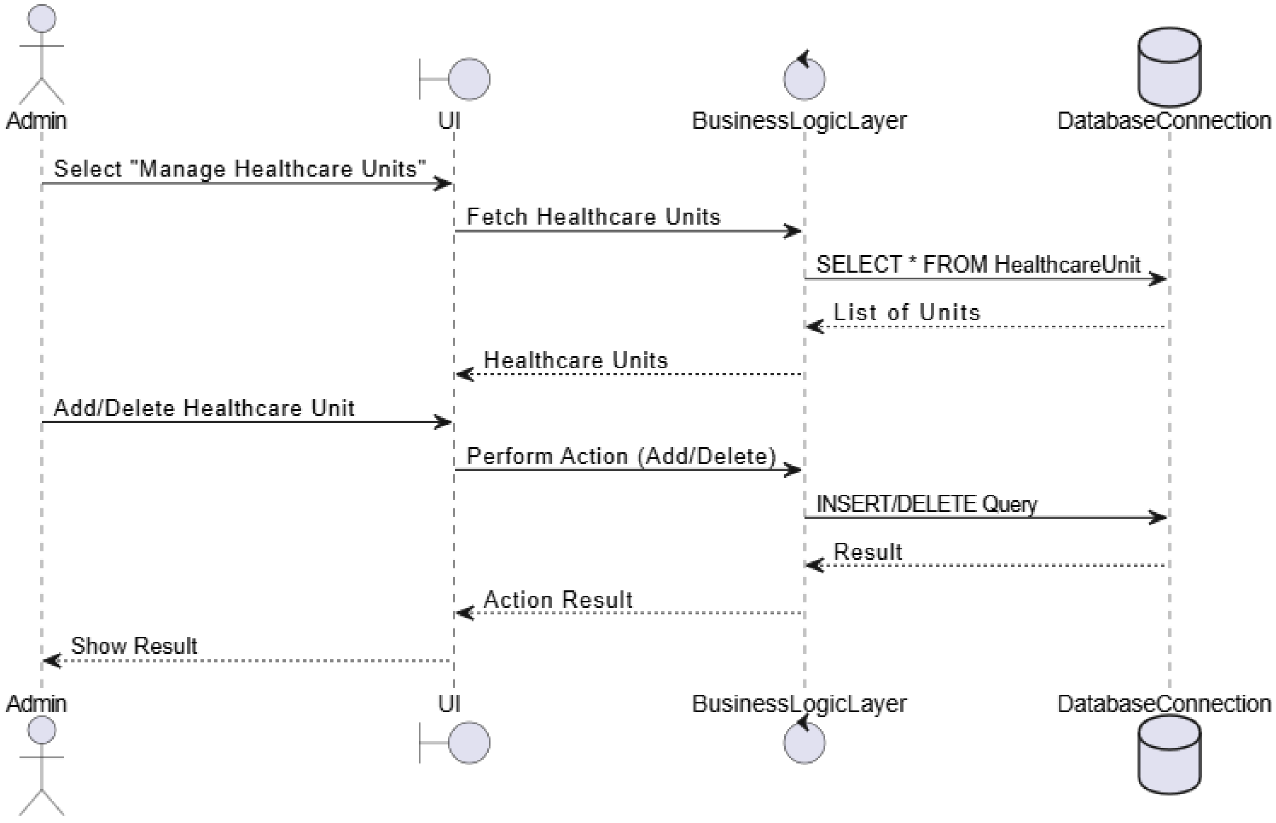
5. System Sequence Diagram

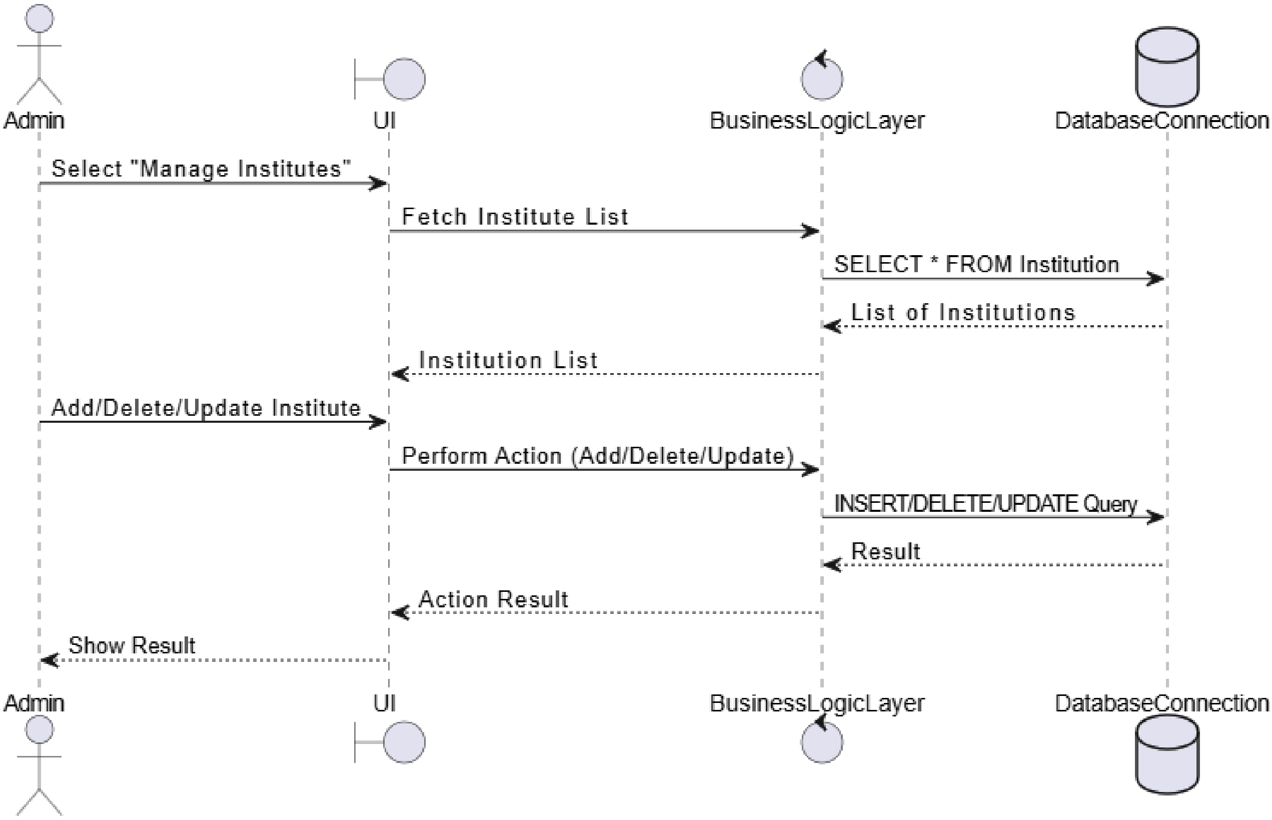

Resident
UI
BusinessLogicLayer
DatabaseConnection
User
Press Button (e.g., Reserve Facility)
create RecreationFacility Object
reserveFacility(recreationFacility)
Boolean (success)
Reservation Result (success or fail)
Show Result
Press Button (e.g., Add Feedback)
create Feedback Object
insertPublicFeedback(feedback)
Boolean (success)
Feedback Submission Result
Show Result
Press Button (e.g., register property)
create registerproperty Object
registration(registerproperty)
Boolean (success)
Registration Result (success or fail)
Show Result
Press Button (e.g., Pay Bills)
create billPayment Object
paybills()billPayment
Boolean (success)
payment Result (success or fail)
Show Result
Press Button (e.g., request public service)
create requestpublicservice Object
requestservice(requestpublicservice)
Boolean (success)
Request Result (success or fail)
Show Result
Press Button (e.g., Vote)
create Voting Object
digitalvoting(vote)
Boolean (success)
Voting Result (success or fail)
Show Result
Press Button (e.g., Acquire Business License)
create Acquiringlicense Object
acquirelicense(acquiringlicense)
Boolean (success)
AcquiringLicense Result (success or fail)
Show Result
Press Button (e.g., Report crime incident)
create reporting Object
reporting(reportingincident)
Boolean (success)
Reporting Result (success or fail)
Show Result
Resident
UI
BusinessLogicLayer
DatabaseConnection
User

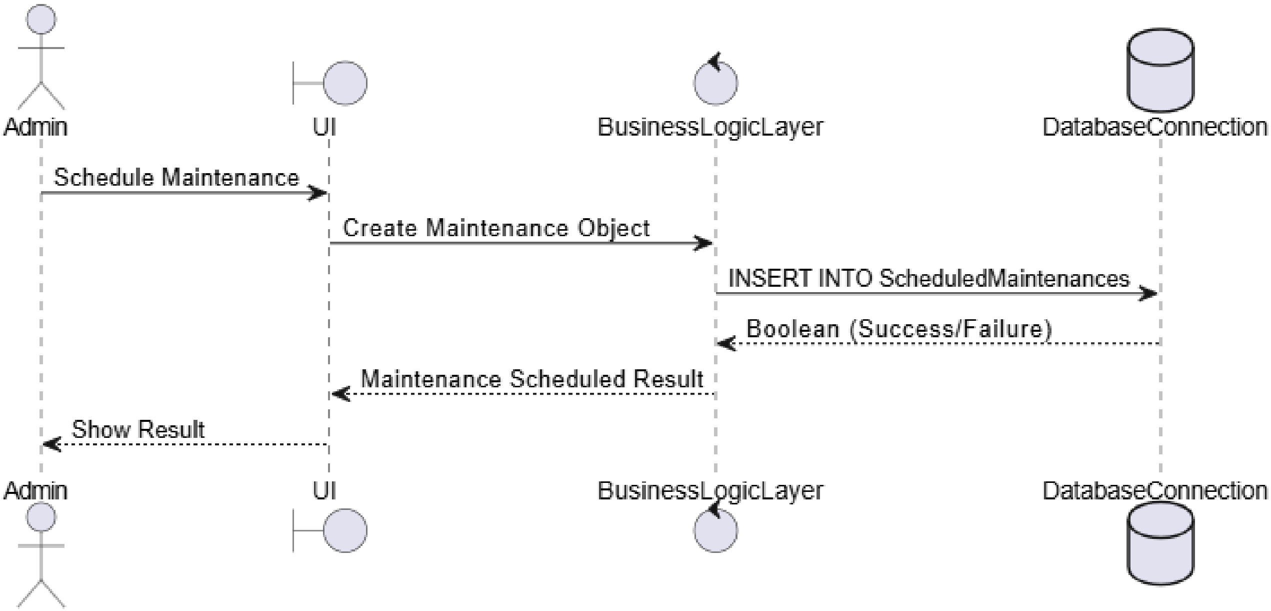

Admin
UI
BusinessLogicLayer
DatabaseConnection
User
Press Button (e.g., ManageInstitutes)
create managinginstitutes Object
manageInstitutes(instituesManagement)
Boolean (success)
isntituteManagement Result (success or fail)
Show Result
Press Button (e.g., ManageHealthCare)
create Feedback Object
managehealthcare(Healthcare)
Boolean (success)
Managing healthcare Result
Show Result
Press Button (e.g., scheduleInfrastructure)
create maintainance Object
scheduling(Infrastructure)
Boolean (success)
schedule Result (success or fail)
Show Result
Admin
UI
BusinessLogicLayer
DatabaseConnection
User

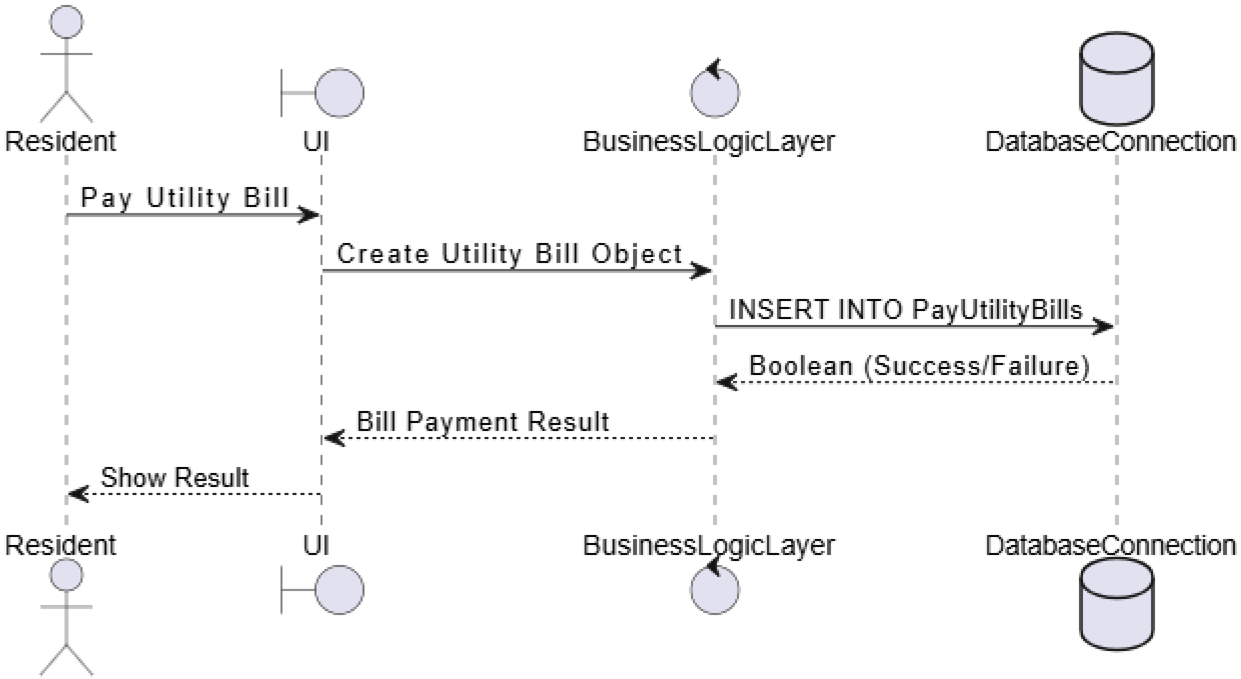
6. Sequence Diagram

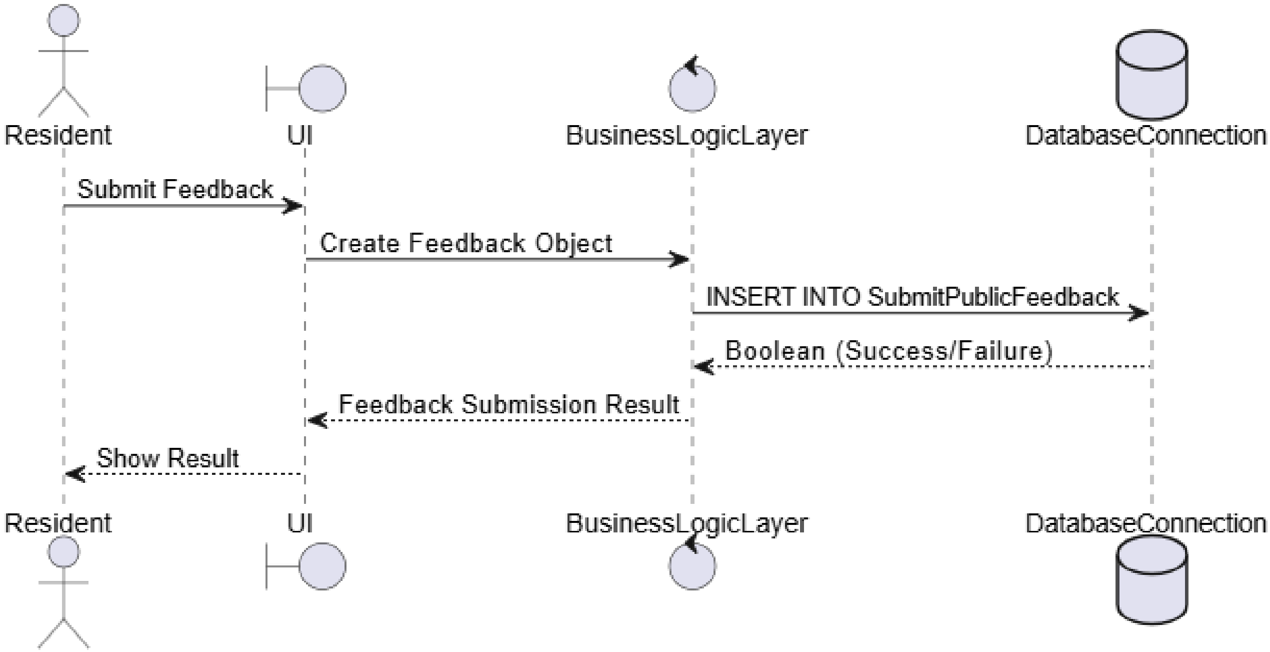


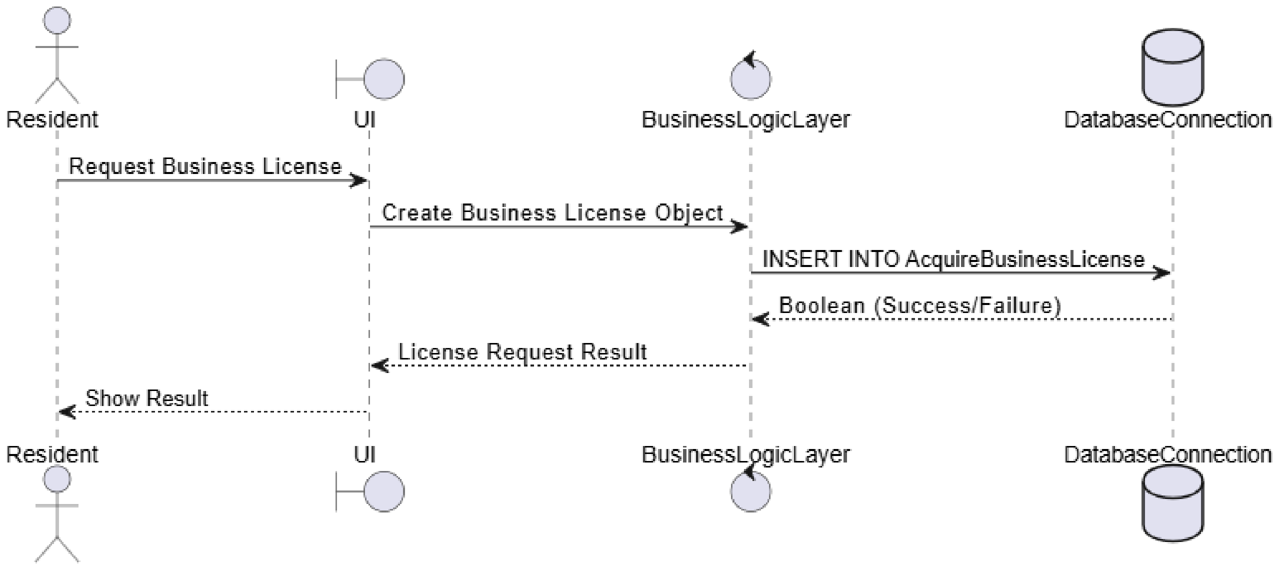


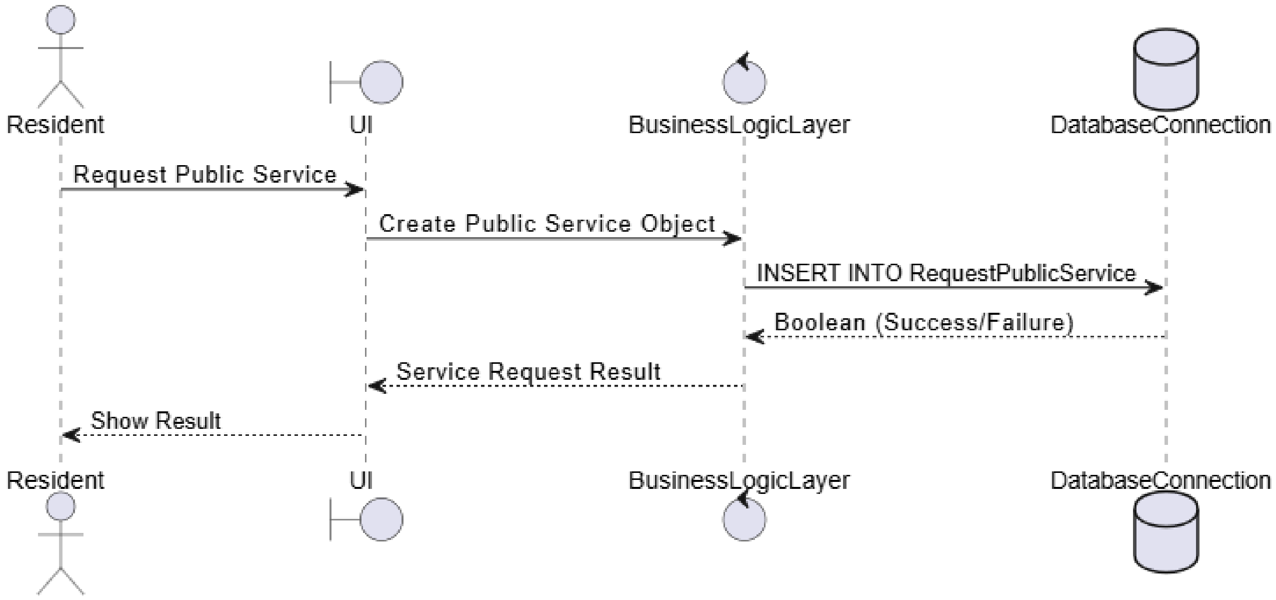

Admin
UI
BusinessLogicLayer
DatabaseConnection
Select "Manage Institutes"
Fetch Institute List
SELECT * FROM Institution
List of Institutions
Institution List
Add/Delete/Update Institute
Perform Action (Add/Delete/Update)
INSERT/DELETE/UPDATE Query
Result
Action Result
Show Result
Admin
UI
BusinessLogicLayer
DatabaseConnection

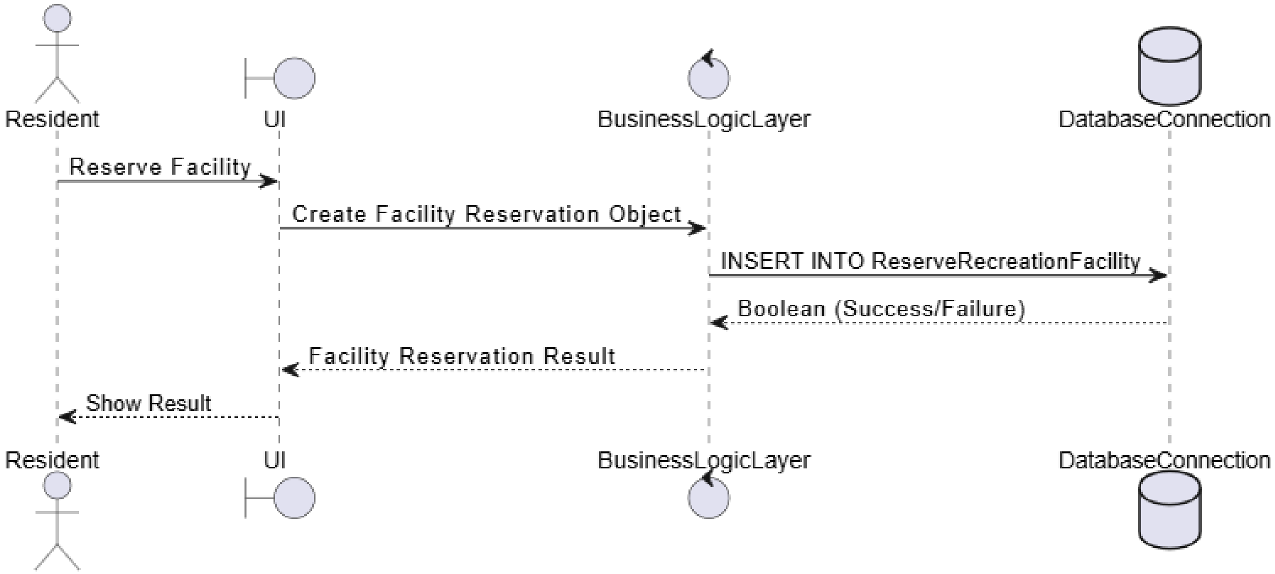

Admin
UI
BusinessLogicLayer
DatabaseConnection
Schedule Maintenance
Create Maintenance Object
INSERT INTO ScheduledMaintenances
Boolean (Success/Failure)
Maintenance Scheduled Result
Show Result
Admin
UI
BusinessLogicLayer
DatabaseConnection

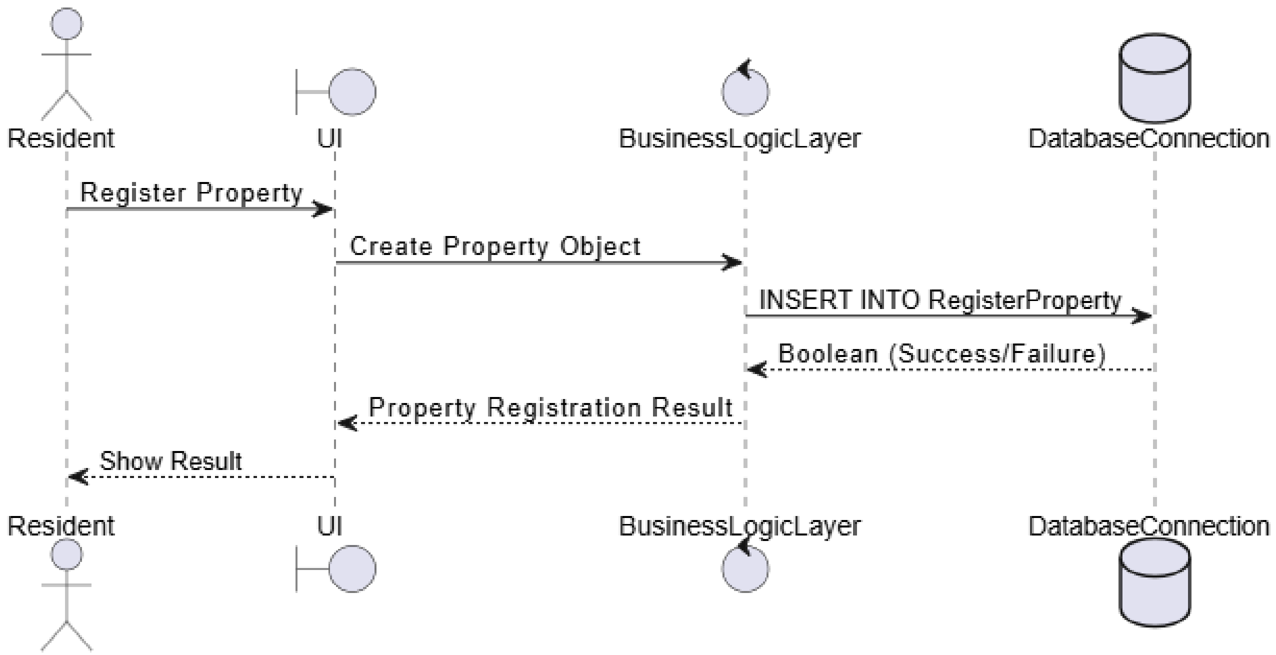

Resident
UI
BusinessLogicLayer
DatabaseConnection
Pay Utility Bill
Create Utility Bill Object
INSERT INTO PayUtilityBills
Boolean (Success/Failure)
Bill Payment Result
Show Result
Resident
UI
BusinessLogicLayer
DatabaseConnection

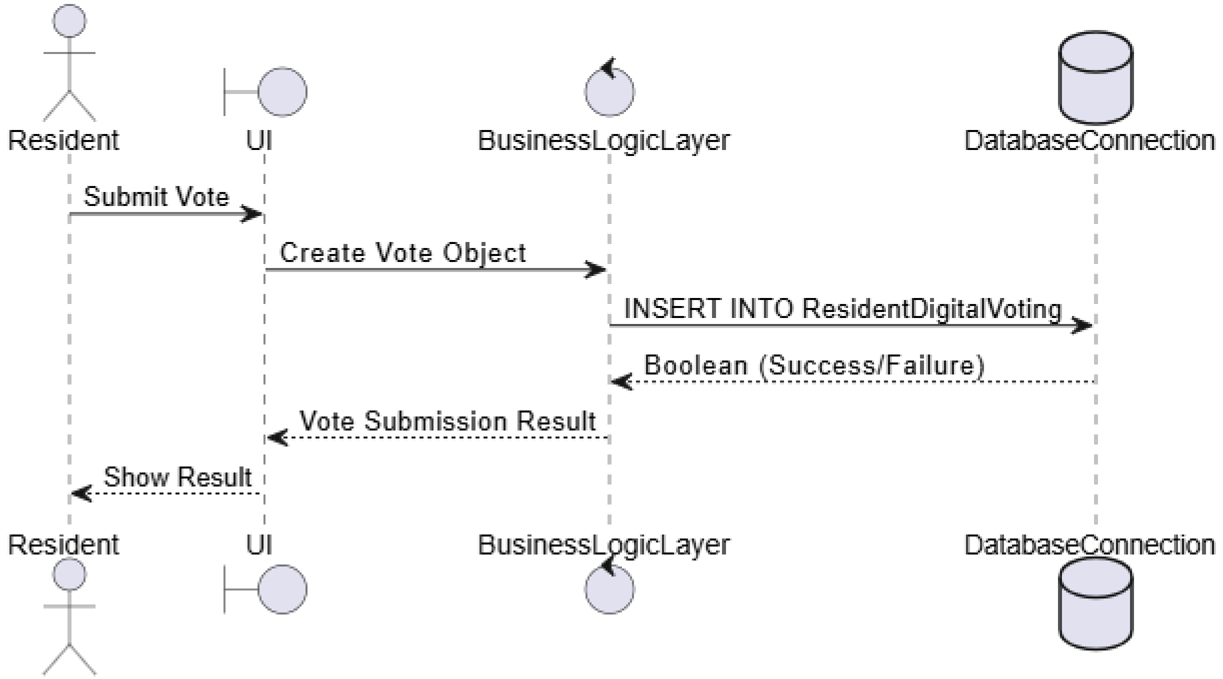

Resident
UI
BusinessLogicLayer
DatabaseConnection
Submit Feedback
Create Feedback Object
INSERT INTO SubmitPublicFeedback
Boolean (Success/Failure)
Feedback Submission Result
Show Result
Resident
UI
BusinessLogicLayer
DatabaseConnection


Resident
UI
BusinessLogicLayer
DatabaseConnection
Request Business License
Create Business License Object
INSERT INTO AcquireBusinessLicense
Boolean (Success/Failure)
License Request Result
Show Result
Resident
UI
BusinessLogicLayer
DatabaseConnection

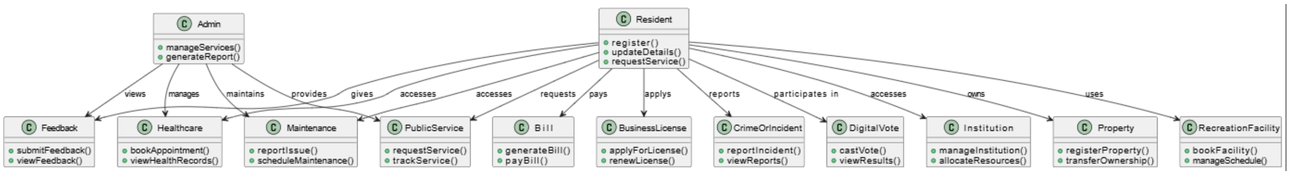

Resident
UI
BusinessLogicLayer
DatabaseConnection
Request Public Service
Create Public Service Object
INSERT INTO RequestPublicService
Boolean (Success/Failure)
Service Request Result
Show Result
Resident
UI
BusinessLogicLayer
DatabaseConnection


Resident
UI
BusinessLogicLayer
DatabaseConnection
Reserve Facility
Create Facility Reservation Object
INSERT INTO ReserveRecreationFacility
Boolean (Success/Failure)
Facility Reservation Result
Show Result
Resident
UI
BusinessLogicLayer
DatabaseConnection

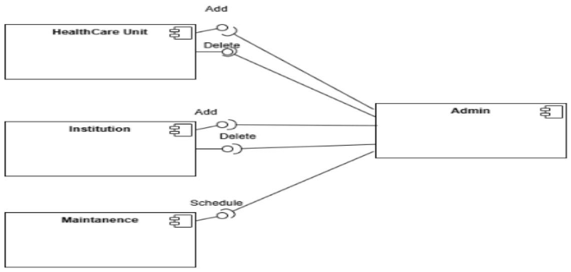


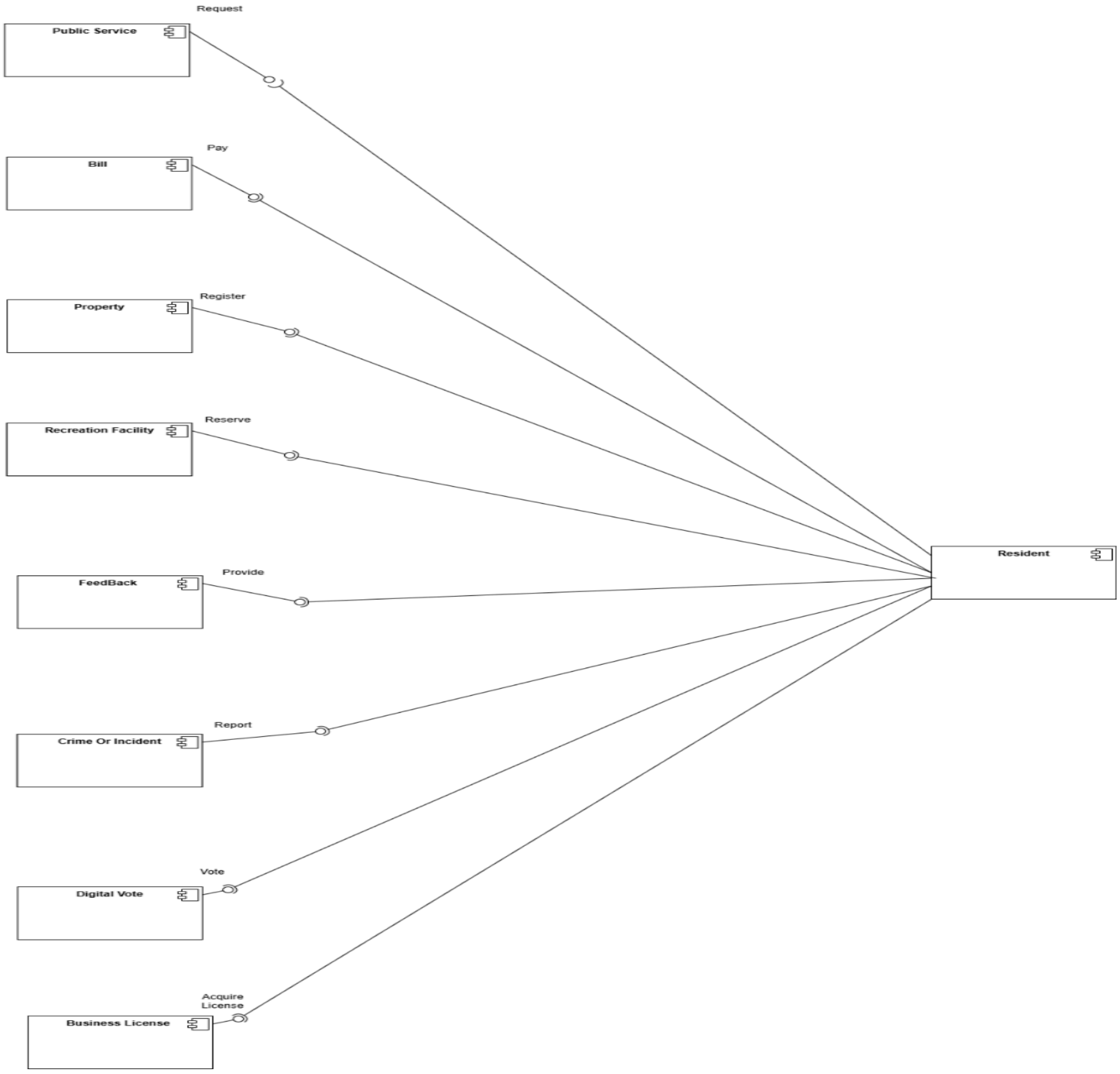


7. Class Diagram

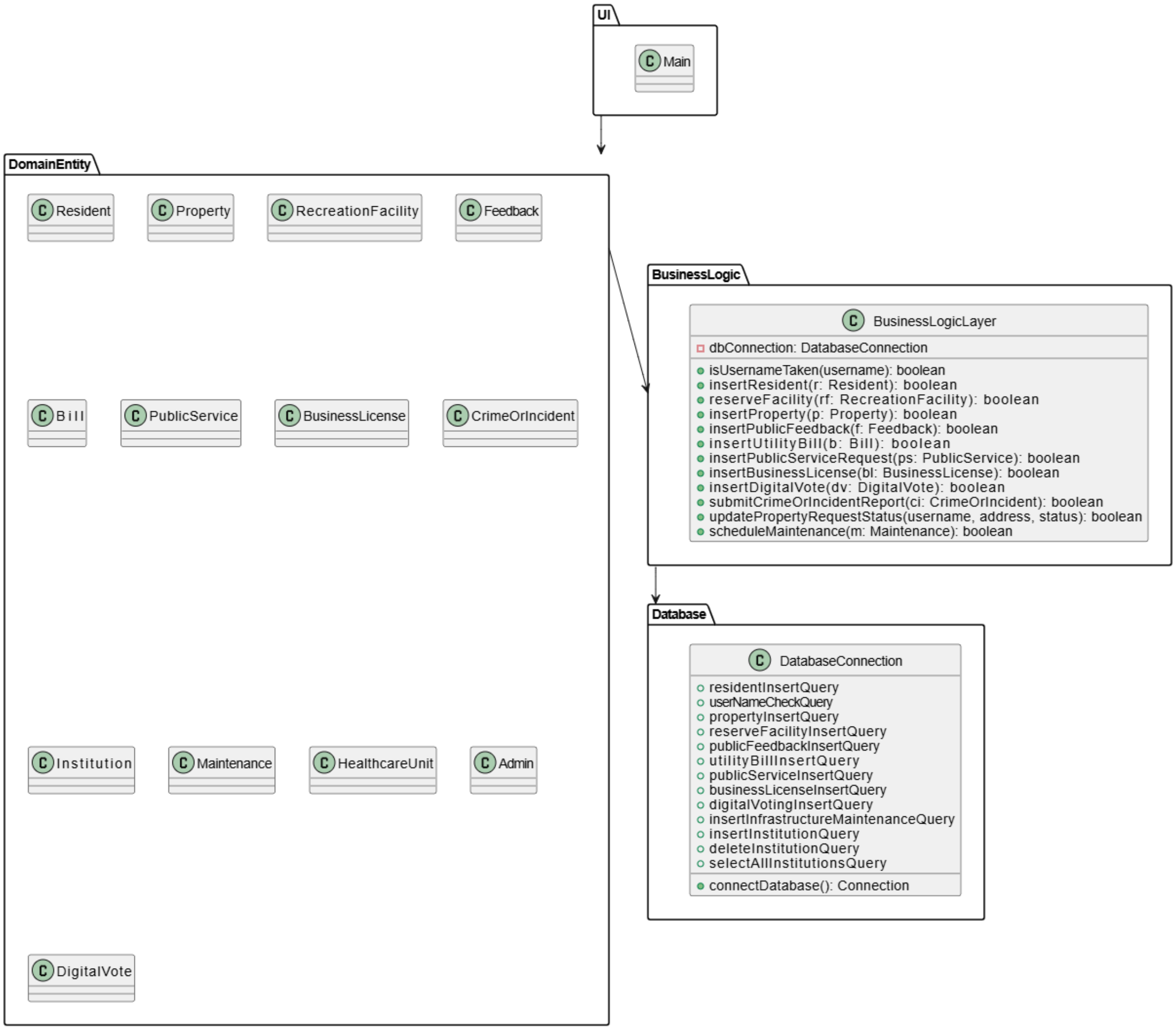


8. Component Diagram

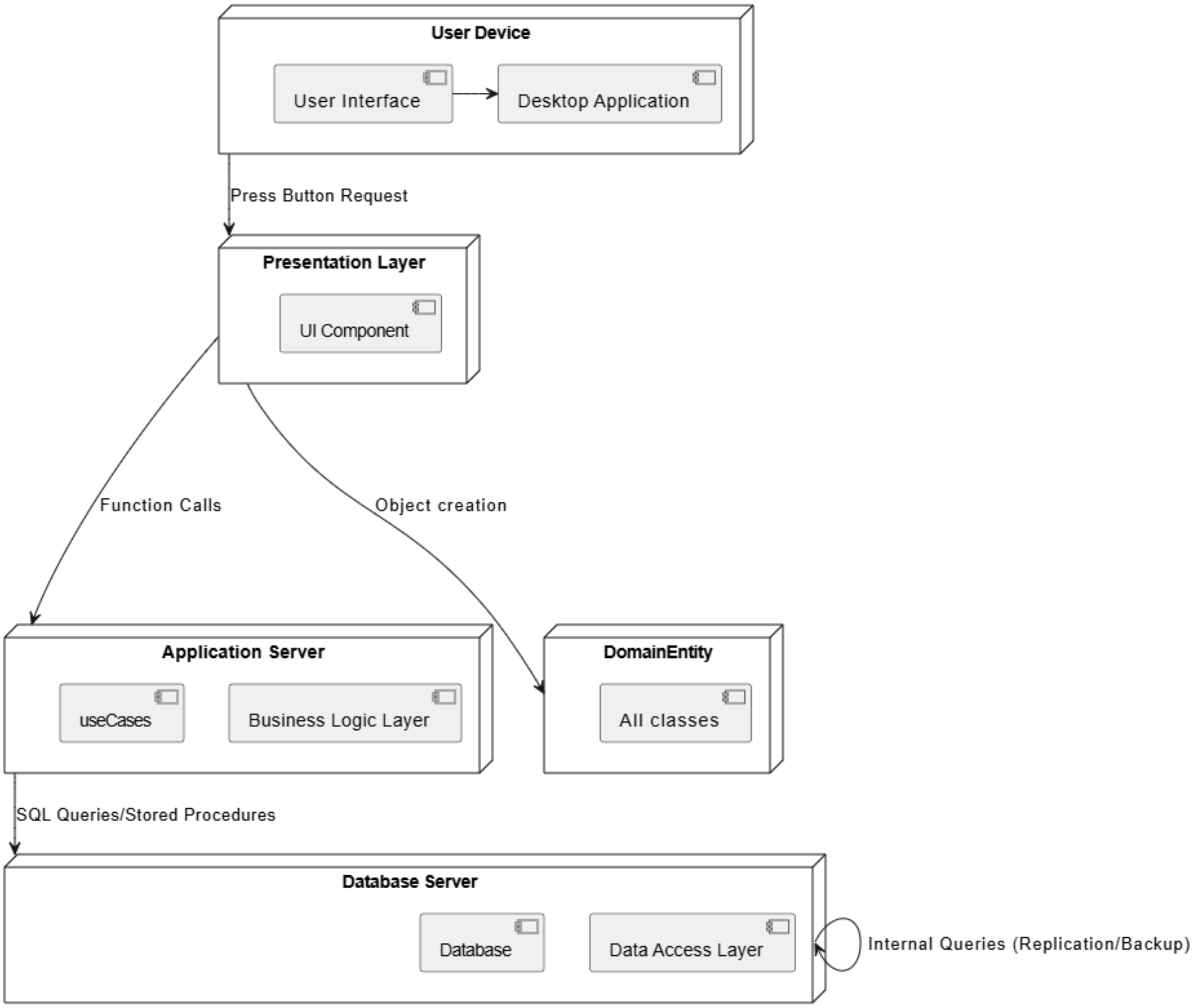




9. Package Diagram


UI
Main
DomainEntity
Resident
Property
RecreationFacility
Feedback
Bill
PublicService
BusinessLicense
CrimeOrIncident
Institution
Maintenance
HealthcareUnit
Admin
DigitalVote
BusinessLogic
BusinessLogicLayer
dbConnection: DatabaseConnection
isUsernameTaken(username): boolean
insertResident(r: Resident): boolean
reserveFacility(rf: RecreationFacility): boolean
insertProperty(p: Property): boolean
insertPublicFeedback(f: Feedback): boolean
insertUtilityBill(b: Bill): boolean
insertPublicServiceRequest(ps: PublicService): boolean
insertBusinessLicense(bl: BusinessLicense): boolean
insertDigitalVote(dv: DigitalVote): boolean
submitCrimeOrIncidentReport(ci: CrimeOrIncident): boolean
updatePropertyRequestStatus(username, address, status): boolean
scheduleMaintenance(m: Maintenance): boolean
Database
DatabaseConnection
residentInsertQuery
userNameCheckQuery
propertyInsertQuery
reserveFacilityInsertQuery
publicFeedbackInsertQuery
utilityBillInsertQuery
publicServiceInsertQuery
businessLicenseInsertQuery
digitalVotingInsertQuery
insertInfrastructureMaintenanceQuery
insertInstitutionQuery
deleteInstitutionQuery
selectAllInstitutionsQuery
connectDatabase(): Connection

10. Deployment Diagram


User Device
User Interface
Desktop Application
Press Button Request
Presentation Layer
UI Component
Function Calls
Object creation
Application Server
useCases
Business Logic Layer
DomainEntity
All classes
SQL Queries/Stored Procedures
Database Server
Database
Data Access Layer
Internal Queries (Replication/Backup)